



Technologia i rozwiązania

JBoss AS 7

Tworzenie aplikacji

Wykorzystaj potencjał serwera aplikacji!

Helion



Francesco Marchioni

[PACKT]
PUBLISHING

Tytuł oryginału: JBoss AS 7 Development

Tłumaczenie: Rafał Jońca

ISBN: 978-83-246-8664-3

Copyright © Packt Publishing 2013.

First published in the English language under the title „JBoss AS 7 Development”.

Polish edition copyright © 2014 by Helion S.A.

All rights reserved.

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/jboas7>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

Spis treści

O autorze	9
<hr/>	
O recenzentach	11
<hr/>	
Wstęp	13
<hr/>	
Zawartość tej książki	13
Co jest potrzebne przy czytaniu książki?	14
Do kogo kierowana jest książka?	15
Konwencje stosowane w książce	15
Pobranie przykładów dla książki	16
Errata	16
Piractwo	16
<hr/>	
Rozdział 1. Zaczynamy przygodę z JBoss AS 7	17
<hr/>	
Krótkie omówienie Javy EE i JBoss AS 7	17
Witamy w Javie EE 6	18
Nowe funkcje wprowadzone w JBoss AS 7	20
Instalacja serwera i komponentów klienta	21
Instalacja Javy SE	22
Instalacja środowiska Eclipse	26
Alternatywne środowiska programistyczne	28
Instalacja narzędzia Maven	29
Podsumowanie	30
<hr/>	
Rozdział 2. Nowości w JBoss AS 7	31
<hr/>	
Podstawowe koncepcje AS 7	31
System plików AS 7	33
Zarządzanie serwerem aplikacji	36
Zarządzanie JBoss AS 7 przy użyciu interfejsu webowego	36
Uruchomienie konsoli webowej	37

Wdrożenie pierwszej aplikacji na serwerze JBoss AS 7	39
Zaawansowane opcje wdrożenia w Eclipse	42
Zarządzanie wdrożeniami z poziomu konsoli webowej	43
Wdrażanie aplikacji przy użyciu narzędzia CLI	46
Podsumowanie	48
Rozdział 3. Wprowadzenie do Javy EE 6 — komponenty EJB	49
EJB 3.1 — nowe funkcjonalności	49
Tworzenie singletonowych komponentów EJB	50
Konfiguracja pliku pom.xml	54
Tworzenie kodu aplikacji EJB	55
Sterowanie współbieżnością ziarna	57
Przygotowanie ziaren sesyjnych	58
Dodanie ziarna bezstanowego	59
Dodanie ziarna sesyjnego	60
Wdrożenie aplikacji EJB	61
Tworzenie zdalnego klienta EJB	64
Konfiguracja pliku pom.xml projektu klienta	67
Tworzenie kodu klienta EJB	68
Uruchomienie aplikacji klienckiej	71
Użycie usługi czasomierza EJB	75
Dodanie do komponentu EJB metod asynchronicznych	77
Podsumowanie	81
Rozdział 4. Poznawanie CDI	83
Wprowadzenie do CDI	83
Ziarna nazwane	85
Zakresy CDI	86
Implementacja CDI w JBoss AS	87
Przekształcenie systemu rezerwacji biletów	88
Tworzenie kodu ziaren	91
Czy komponenty EJB i ziarna zarządzane przez JSF są już przestarzałe?	107
Podsumowanie	107
Rozdział 5. Łączenie trwałości z CDI	109
Trwałość danych i standardy	109
Korzystanie z JPA	110
Dodanie trwałości do aplikacji	111
Konfiguracja bazy danych	111
Instalacja sterownika JDBC w JBoss AS 7	112
Tworzenie projektu Maven	114
Dodanie konfiguracji Maven	115
Tworzenie encji	116
Dodanie walidacji ziarna	118
Konfiguracja trwałości	119
Dodanie klas produkujących	120
Tworzenie kodu zapytań	123

Dodanie do aplikacji usług	123
Dodanie kontrolera sterującego zadaniami użytkowników	126
Tworzenie widoków JSF	128
Uruchomienie przykładu	132
Podsumowanie	134
Rozdział 6. Testowanie aplikacji	135
Testy jednostkowe i integracyjne	135
Narzędzia pomagające w testach	136
Korzystanie z narzędzia Arquillian	137
Pisanie testu Arquillian	137
Konfiguracja pliku pom.xml	139
Napisanie pierwszego testu	141
Uruchomienie testu TicketTest	143
Uruchomienie testu w zarządzanym kontenerze	144
Rozbudowa testu	145
Informacje dodatkowe	148
Podsumowanie	148
Rozdział 7. Tworzenie aplikacji wykorzystujących JBoss JMS Provider	149
Krótkie wprowadzenie do JMS	150
Elementy składowe JMS	151
Podsystem komunikatów w JBoss	152
Tworzenie i wykorzystanie fabryk połączeń	153
Użycie celów JMS	155
Dodanie do aplikacji ziaren sterowanych komunikatami	156
Użycie JMS do integracji z innymi systemami	165
Przykład z życia wzięty — integracja HornetQ i ActiveMQ	165
Podsumowanie	169
Rozdział 8. Dodanie do aplikacji usług sieciowych	171
Tworzenie usług sieciowych bazujących na SOAP	172
Strategie tworzenia usług sieciowych typu SOAP	172
Stos usług sieciowych SOAP w JBoss	173
Krótki przegląd architektury JAX-WS	174
Tworzenie usługi sieciowej w JBoss AS 7	175
Tworzenie usługi sieciowej bazującej na REST	185
Dostęp do zasobów typu REST	186
Usługa sieciowa typu REST w JBoss	187
Wybór między usługami REST i SOAP	193
Podsumowanie	193
Rozdział 9. Zarządzanie serwerem aplikacji	195
Wprowadzenie do interfejsu wiersza poleceń (CLI)	195
Uruchomienie wiersza poleceń	196
Konstrukcja poleceń CLI	197

Wdrażanie aplikacji przy użyciu CLI	201
Tworzenie skryptów CLI	203
Użycie zaawansowanych języków do tworzenia wyrafinowanych skryptów CLI	205
Użycie języków skryptowych do wykonywania operacji na CLI	206
Bezpośrednie użycie API zarządzania do sterowania serwerem aplikacji	209
Odczytywanie opisów modelu zarządzania za pomocą API bezpośredniego	209
Podsumowanie	212
Rozdział 10. Klastry aplikacji JBoss AS 7	213
Podstawy wiedzy o klastrach	213
Klastry w JBoss AS 7	214
Uruchamianie klastra węzłów samodzielnych	215
Uruchamianie klastra węzłów domenowych	216
Wdrażanie aplikacji klastrowych	220
Klastry EJB	221
Tworzenie klastrów aplikacji webowych	231
Równoważenie obciążenia w aplikacjach webowych	232
Podsumowanie	238
Rozdział 11. Bezpieczeństwo aplikacji JBoss AS 7	239
API bezpieczeństwa w języku Java	239
Podsystem bezpieczeństwa JBoss AS 7	241
Konfiguracja pierwszego modułu logowania	242
Użycie modułu logowania w aplikacji systemu rezerwacji biletów	243
Przełączenie na bezpieczeństwo bazujące na formularzu	245
Tworzenie modułu logowania wykorzystującego bazę danych	246
Zabezpieczenie komponentów EJB	249
Zabezpieczanie warstwy transportowej	252
Uruchamianie komunikacji SSL w JBoss AS	255
Podsumowanie	263
Dodatek A. Szybkie tworzenie aplikacji przy użyciu JBoss Forge	265
Instalacja Forge	265
Uruchomienie Forge	266
Tworzenie pierwszej aplikacji Javy EE 6 w JBoss Forge	268
Budowanie i wdrożenie aplikacji	271
Aplikacja forge-demo w akcji	272
Skorowidz	275

Testowanie aplikacji

Przykładowa aplikacja zawiera już dosyć dobrą mieszankę technologii — są w niej najbardziej kluczowe elementy Javy EE (poza systemem Java Messaging, który zostanie opisany w następnym rozdziale). Jednym z zadań programistów, poza tworzeniem aplikacji, jest również wykonywanie testów aplikacji wdrożonej lub uruchomionej zdalnie na serwerze. W tym rozdziale omówimy framework JBoss AS o nazwie **Arquillian**, który stara się zasłużyć na miano standardowego frameworka testów integracyjnych aplikacji biznesowych.

Oto tematy omówione w tym rozdziale.

- Wprowadzenie do testów aplikacji biznesowych — od obiektów naśladujących do frameworka Arquillian.
- Integracja przypadków testowych narzędzia Arquillian z aplikacją rezerwacji biletów.
- Wykorzystanie IDE Eclipse oraz konsoli Maven do uruchamiania testów Arquillian.

Testy jednostkowe i integracyjne

Słowo „testy” można interpretować na wiele sposobów; najczęściej oznacza ono weryfikację podstawowych funkcjonalności aplikacji. Testy można jednak przeprowadzać w różny sposób, w zależności od tego, co jest obiektem weryfikacji, a także w jakim środowisku prowadzi się weryfikację.

Najpowszechniejszym rodzajem testów jest tak zwany **test jednostkowy**, tworzony przez programistę, by sprawdzić, czy pewien niewielki, dobrze wyizolowany fragment aplikacji wykonuje swoje zadanie zgodnie z przeznaczeniem. Testy jednostkowe mają bardzo wąski zasięg; można je łatwo napisać i wykonać, a ich użyteczność w dużej mierze zależy od tego, co programista uzna za wartę sprawdzenia. Testy tego rodzaju są definiowane tylko przez programistę i nie są w sposób bezpośredni użyteczne dla innych osób. Oczywiście, jeśli programista

wykonuje swoją pracę właściwie, na ich stosowaniu skorzystają zarówno testerzy, jak i zwykli użytkownicy, bo ryzyko błędów zmniejszy się znacząco.

Bardziej wyrafinowanym rodzajem testu jest **test integracyjny**. Taki test ma za zadanie sprawdzić współpracę różnych elementów systemu; ponieważ obejmuje swym zasięgiem całą aplikację, wymaga znacznie większego nakładu pracy. Najczęściej niezbędne jest przygotowanie bazy danych i sprzętu, na którym taki test będzie uruchamiany. Testy integracyjne w zdecydowanie bardziej przekonujący sposób sprawdzają działanie systemu jako całości (szczególnie z punktu widzenia osób niezajmujących się programowaniem), jeśli tylko środowisko testowe przypomina środowisko produkcyjne.

Narzędzia pomagające w testach

Jak nietrudno sobie wyobrazić, w każdym rodzaju testu używa się innego podejścia. W testach jednostkowych najpopularniejszymi elementami pomocniczymi są **obiekty naśladujące**. Jeśli obiekt poddawany testowi posiada metody, a metody te zależą od innych obiektów, jako zależność można przekazać spreparowany obiekt naśladujący oryginalną zależność. Dzięki temu test można przeprowadzić w izolacji.

Doskonałym przykładem takiej sytuacji są aplikacje MVC, w których występuje warstwa **doępu do danych DAO (Data Access Object)** i kontroler zapewniający logikę biznesową. Gdy chcemy przetestować kontroler uzależniony do warstwy DAO, wystarczy utworzyć obiekt naśladujący obiekt DAO i przekazać go do kontrolera.

Takie podejście, choć szybkie w realizacji i łatwe do zrozumienia, ma kilka ograniczeń. Po pierwsze, integracja dotyczy sztucznego środowiska, w którym można poczynić niewłaściwe założenia na temat zachowania i stabilności rzeczywistego środowiska.

Po drugie, uzyskuje się trudną w utrzymaniu bibliotekę obiektów naśladujących, która daje to przyjemne uczucie dobrze spełnionego obowiązku, gdy wszystkie testy działają poprawnie.

Choć obiekty naśladujące mają swoje zalety (szczególnie na etapie wstępnych prac nad systemem, gdy nie istnieje jeszcze pełna implementacja podsystemu), najlepiej pozostać tak blisko, jak to możliwe, docelowego środowiska, w którym działać ma finalna wersja kodu.

Arquillian to platforma ułatwiająca tworzenie testów integracyjnych dla aplikacji Javy. Odpowiada za przygotowanie kontenera, zarządzanie nim, a także wdrożeniem go, oraz za inicjalizację frameworka, by programista mógł się skupić na pisaniu testów — prawdziwych testów. Arquillian minimalizuje nakład pracy programisty związany z przygotowaniem środowiska testowego, bo zajmuje się między innymi:

- zarządzaniem cyklem życia kontenera (jego uruchomieniem i zatrzymaniem),
- łączeniem klas testów z klasami zależności i zasobami w jedno archiwum do natychmiastowego wdrożenia,

- rozbudowywaniem klas testów (na przykład przez automatyczne rozwiązanie wstrzyknięć @Inject, @EJB i @Resource),
- wdrożeniem archiwum na serwerze testowym (wdrożeniem i usunięciem wdrożenia), a także przechwyceniem wyników testu.

W następnym podrozdziale wskażemy elementy niezbędne do uruchomienia testów integracyjnych przy użyciu frameworka Arquillian.

Korzystanie z narzędzia Arquillian

Choć Arquillian nie jest zależne od konkretnego narzędzia do budowania pakietów, najczęściej stosuje się je w połączeniu z narzędziem Maven, które zarządza zależnościami, więc ułatwia zadanie dołączenia bibliotek Arquillian do aplikacji, bo znajdują się w centralnym repozytorium Maven.

W zależności od archetypu użytego do wygenerowania projektu, może istnieć różna struktura folderów, ale nie jest to żadną przeszkodą. Istotne jest tylko, by w folderze *src* znalazła się następująca struktura:

- *main/*
 - *java/* — tu trzeba umieścić wszystkie pliki źródłowe aplikacji Java (w odpowiednich pakietach),
 - *resources/* — tu trzeba umieścić wszystkie pliki konfiguracyjne aplikacji,
- *test/*
 - *java/* — tu trzeba umieścić wszystkie pliki źródłowe testów w języku Java (w odpowiednich pakietach),
 - *resources/* — tu trzeba umieścić wszystkie pliki konfiguracyjne testów (na przykład *persistence.xml*).

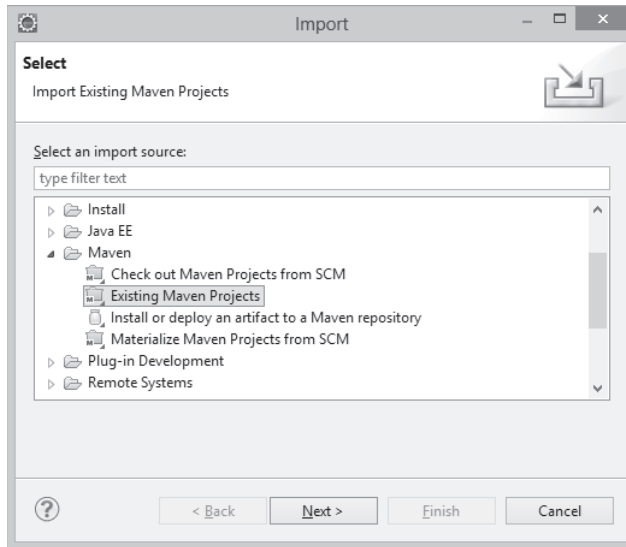
Większość prac odbywa się w folderze *test/java*, który zawiera klasy testów Arquillian.

Pisanie testu Arquillian

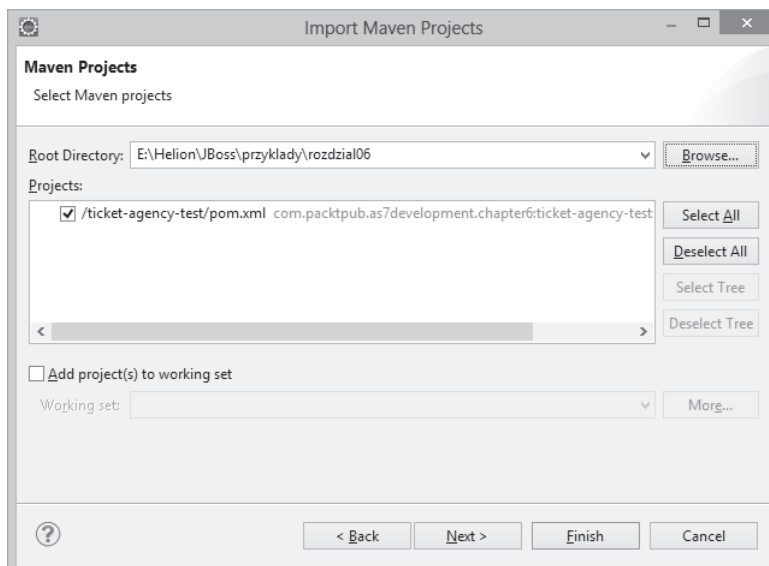
Jeśli kiedykolwiek tworzyłeś testy przy użyciu JUnit (<http://www.junit.org>), testy Arquillian będą wyglądały bardzo podobnie, ale z kilkoma istotnymi dodatkami.

Do przygotowania testów, podobnie jak wcześniej, wykorzystamy Eclipse i Maven. Jeśli do projektu chce się dodać klasy testów, nie trzeba tworzyć nowego projektu. By jednak zapewnić odpowiedni proces nauki, przykład znajduje się w osobnym projekcie, więc będzie wiadomo, co dokładnie dodać, by uruchomić testy Arquillian.

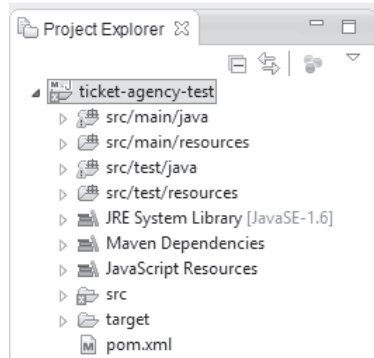
Aby uniknąć ponownego tworzenia całego projektu od podstaw, można po prostu sklonować projekt `ticket-agency-jpa`, nadać mu nazwę `ticket-agency-test` i przenieść główny pakiet z `com.packtpub.as7development.chapter5` na `com.packtpub.as7development.chapter6`. Jeśli to nadal za dużo pracy, można również zaimportować projekt *rozdział06* z archiwum przykładów dotyczących książki. Z menu *File* wybieramy polecenie *Import*, a następnie *Existing Maven Projects*.



Wskazujemy folder zawierający przykłady dla rozdziału 6., a następnie zaznaczamy na liście plik `pom.xml`.



Klikamy przycisk *Finish* i sprawdzamy, czy w widoku *Project Explorer* struktura folderów odpowiada poniższej.



Przyjrzyjmy się teraz podstawowym zależnościom wymaganych do uruchomienia przypadków testowych Arquillian.

Konfiguracja pliku pom.xml

Pierwszą zależnością niezbędną do dodania w celu uruchomienia testów Arquillian jest zależność junit.

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <scope>test</scope>
</dependency>
```

Warto zwrócić uwagę, że w zależności tej (podobnie jak w kolejnych) stosuje się zakres test. Oznacza to, że Maven użyje jej tylko do uruchamiania testów i nie dołączy do końcowej wersji artefaktów.

Zależność junit stanowi część **BOM (Bill of Materials)** z JBoss, więc nie trzeba wskazywać wersji zależności. To samo dotyczy zależności związanych z Arquillian.

Aby testować funkcje typu enterprise, takie jak EJB lub **JTA (Java Transaction API)**, konieczne jest dołączenie zależności `org.jboss.arquillian.junit`.

```
<dependency>
  <groupId>org.jboss.arquillian.junit</groupId>
  <artifactId>arquillian-junit-container</artifactId>
  <scope>test</scope>
</dependency>
```

Ponieważ w testach Arquillian używa się specjalnego protokołu do komunikacji z serwerem aplikacji, konieczne jest dodanie zależności `org.jboss.arquillian.protocol` (jak wskazuje nazwa, jest zgodna ze specyfikacją Servlet 2.5 i 3.0).

```
<dependency>
  <groupId>org.jboss.arquillian.protocol</groupId>
  <artifactId>arquillian-protocol-servlet</artifactId>
  <scope>test</scope>
</dependency>
```

Ostatnią zależnością do dodania jest kontener weld. Musimy go dodać, ponieważ będziemy testować pewne zaawansowane elementy CDI, takie jak komunikacja o zakresie konwersacji.

```
<dependency>
  <groupId>org.jboss.weld</groupId>
  <artifactId>weld-core-test-arquillian</artifactId>
  <version>1.1.9.Final</version>
</dependency>
```

Po ustaleniu zależności do konfiguracji musimy jeszcze dodać dwa profile. Profil służy do utworzenia innego środowiska docelowego dla celów Maven. Oto krótki opis obu profili.

- Pierwszy profil nosi nazwę `arq-jbossas-managed`. Uruchomi on nową instancję JBoss AS, wykona testy i wyłączy instancję.

```
<profile>
  <id>arq-jbossas-managed</id>
  <dependencies>
    <dependency>
      <groupId>org.jboss.as</groupId>
      <artifactId>jboss-as-arquillian-containermanaged</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>
</profile>
```

- Drugi profil nosi nazwę `arq-jbossas-remote` i wykonuje testy, korzystając ze zdalnej instancji JBoss AS.

```
<profile>
  <id>arq-jbossas-remote</id>
  <dependencies>
    <dependency>
      <groupId>org.jboss.as</groupId>
      <artifactId>jboss-as-arquillian-containerremote</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>
</profile>
```

Napisanie pierwszego testu

Po zakończeniu konfiguracji można przystąpić do pisania kodu testu. Tworzymy klasę o nazwie `TicketTest` w pakiecie `com.packtpub.as7development.chapter6.test`. Pierwszym elementem, który należy dodać do klasy, jest adnotacja informująca JUnit, że sterowaniem testem powinien zająć się Arquillian.

```
@RunWith(Arquillian.class)
public class TicketTest {
}
```

Arquillian poszukuje w klasie testu metody statycznej oznaczonej adnotacją `@Deployment`, by utworzyć mikrowdrożenie zawierające wszystkie zasoby, podobnie jak miałyby to miejsce w przypadku standardowego wdrożenia za pomocą ulubionych narzędzi.

```
@Deployment
public static Archive<?> createTestArchive() {

    return ShrinkWrap.create(WebArchive.class, "ticket-agency-test.war")
        .addPackage(SeatProducer.class.getPackage())
        .addPackage(Seat.class.getPackage())
        .addPackage(TicketService.class.getPackage())
        .addPackage(DataManager.class.getPackage())
        .addAsResource("META-INF/persistence.xml")
        .addAsWebInfResource(EmptyAsset.INSTANCE, "beans.xml");

}
```

Przedstawione API umożliwiające łatwe tworzenie łańcuchów wywołań zapewnia projekt **ShrinkWrap** (<http://www.jboss.org/shrinkwrap>). Metoda `create` przyjmuje jako parametry typ jednostki wdrożenia (`WebArchive`) i nazwę wdrożenia (w tym przypadku `ticket-agency-test.war`), a następnie umożliwia wskazanie wszystkich zasobów do dołączenia do archiwum. W prezentowanym przykładzie, zamiast dołączać wszystkie klasy, używamy metody pomocniczej `addPackage`, która dodaje wszystkie klasy znajdujące się w wybranym pakiecie (przekazanie do metody wyniku wykonania metody `Seat.class.getPackage()` spowoduje dodanie wszystkich klas zawartych w tym samym pakiecie, co klasa `Seat`).

Po tych wszystkich etapach przygotowawczych możemy dodać metodę przeprowadzającą test.

```
@javax.inject.Inject
TicketService ticketService;

@Inject
Logger log;

@Test
public void testTicketAgency () throws Exception {
```

```

SeatType seatType = new SeatType();
seatType.setDescription("opis");
seatType.setPrice(30);
seatType.setQuantity(5);

ticketService.createSeatType(seatType);
log.info("Utworzono typ miejsc "+seatType.getDescription());
assertNotNull(seatType.getId());
}

```

Metoda `testTicketAgency` utworzy nowy atrybut `SeatType` przy użyciu metody `createSeatType` z klasy `TicketService`. Warto zwrócić uwagę na sposób wstrzyknięcia `TicketService` — odbywa się w ten sam sposób, jakby kod był uruchamiany po stronie serwera.

Zakończyliśmy tworzenie pierwszego przypadku testowego. Musimy jeszcze dodać konfigurację Arquillian, umieszczając ją w pliku `arquillian.xml`, w folderze `src/test/resources`.

```

<?xml version="1.0" encoding="UTF-8"?>

<arquillian xmlns="http://jboss.org/schema/arquillian"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jboss.org/schema/arquillian
    http://jboss.org/schema/arquillian/arquillian_1_0.xsd">

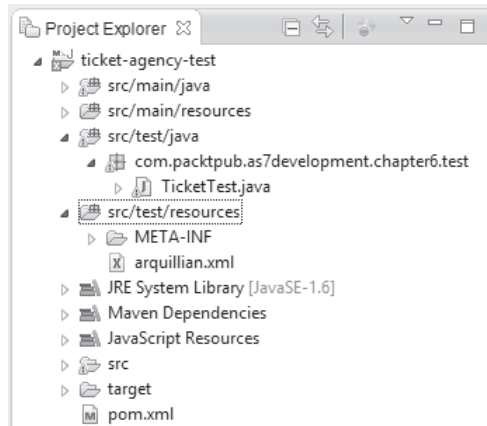
  <defaultProtocol type="Servlet 3.0" />
  <container qualifier="jboss" default="true">
    <configuration>
      <property name="managementAddress">localhost</property>
      <property name="managementPort">9999</property>
    </configuration>
  </container>
</arquillian>

```

Element `defaultProtocol` wymusza na frameworku Arquillian, by stosował protokół Servlet 3.0 dla wszystkich kontenerów. To obecnie najlepiej obsługiwany protokół.

Dodatkowo należy wskazać adres i port systemu zarządzania, jeśli nie jest dostępny pod domyślnymi wartościami (w prezentowanym przykładzie używamy wartości domyślnych, ale dołączyliśmy elementy, by wskazać ich prawidłową lokalizację w pliku konfiguracyjnym).

Przed uruchomieniem testu należy się upewnić, że wszystkie elementy projektu znajdują się we właściwych miejscach.

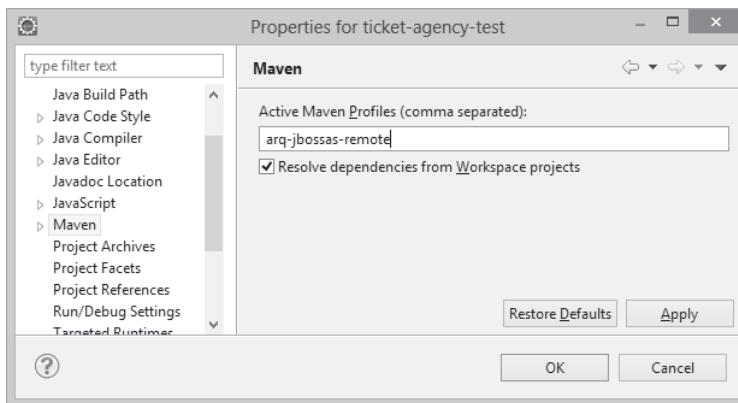


Uruchomienie testu TicketTest

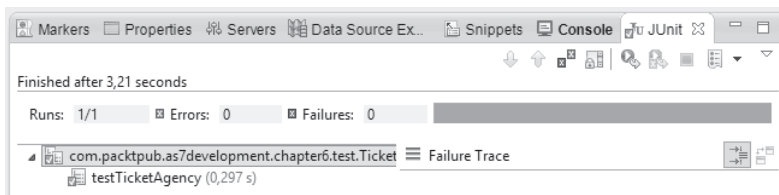
Aby osiągnąć właściwy rezultat, warto wyczyścić i ponownie skompilować cały projekt, by usunąć ewentualne artefakty pochodzące z wcześniejszego projektu *ticket-agency-jpa* (jeśli pliki były kopiowane). Z tego względu trzeba użyć przedstawionego poniżej celu Maven, wykorzystując w tym celu wiersz poleceń lub IDE Eclipse.

```
mvn clean install
```

Następnie musimy wskazać, czy testy mają być uruchamiane przy użyciu profilu zdalnego Arquillian (wykorzystana zostanie istniejąca instancja JBoss AS 7), czy też ma zostać uruchomiona nowa instancja JBoss wraz z zarządzanym profilem. Jeśli serwer JBoss jest już uruchomiony, przechodzimy do właściwości projektu, a potem do właściwości Maven. W polu *Active Maven Profiles* wpisujemy wartość `arq-jbossas-remote`, która została wcześniej zadeklarowana w pliku *pom.xml*.



Teraz wystarczy już tylko kliknąć klasę *TicketTest* w oknie drzewa projektu prawym przyciskiem myszy i wybrać polecenie *Run As/JUnit Test*. Framework Arquillian uruchomi się i wyświetli wynik testu w widoku *JUnit* (jeśli widok nie jest widoczny, wyświetlamy go poleceniem *Window/Show View/JUnit* z menu Eclipse).



Gratulacje! Konsola JUnit informuje o poprawnym wykonaniu pierwszego testu.

Test wykonany poprawnie, ale gdzie znajdują się logi?

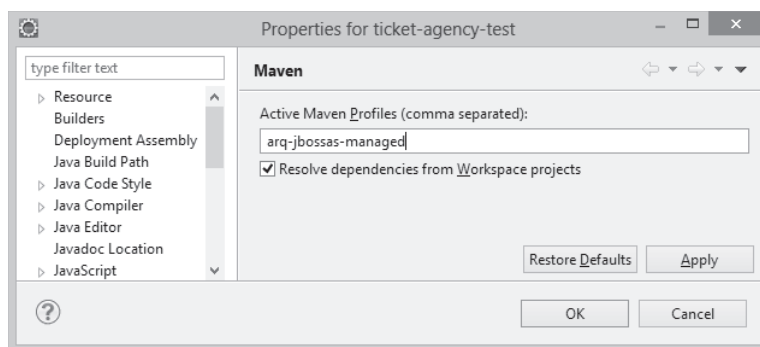
Uruchamiając testy na zdalnym serwerze, zastanawiamy się, dlaczego nie widać wyników wykonania poleceń `System.out` lub `log`. Wynika to z faktu, iż test nie działa na tej samej maszynie wirtualnej Javy, co system uruchamiania testów. Po stronie serwera aplikacji powinny pojawić się w konsoli komunikaty:

```
17:40:14,728 INFO [com.packtpub.as7development.chapter6.test.TicketTest]
↳(http--127.0.0.1-8080-2) Utworzono typ miejsc Balkon
```

Uruchomienie testu w zarządzanym kontenerze

Kiedy używamy kontenera zdalnego (na przykład `arq-jbossas-remote`), Arquillian musi jedynie otrzymać informację, że kontener jest uruchomiony. Komunikacja odbywa się przy użyciu instrukcji sterujących.

Drugie z rozwiązań polega na użyciu zarządzanego kontenera. W tym celu należy zmienić profil Maven na zgodny z poniższym zrzutem ekranu.



Dodatkowo do narzędzia Arquillian trzeba przekazać informację, gdzie został zainstalowany serwer JBoss AS 7, by go uruchomić. Lokalizację głównego folderu serwera definiuje się w pliku *arquillian.xml*.

```
<?xml version="1.0" encoding="UTF-8"?>

<arquillian xmlns="http://jboss.org/schema/arquillian"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jboss.org/schema/arquillian
    http://jboss.org/schema/arquillian/arquillian_1_0.xsd">

  <defaultProtocol type="Servlet 3.0" />
  <container qualifier="jboss" default="true">
    <configuration>
      <property name="jbossHome">C:\jboss-as-7.1.1.Final</property>
    </configuration>
  </container>
</arquillian>
```

To wszystko. Uruchomienie testu spowoduje automatyczne włączenie instancji serwera JBoss AS 7 i wdrożenie na nim aplikacji. Po wykonaniu testu instancja zostanie wyłączona w sposób automatyczny.

Rozbudowa testu

Łatwo zauważyć, że przedstawiony wcześniej kod testu celowo nie był kompletny. Nie przeprowadziliśmy wszystkich niezbędnych operacji, takich jak utworzenie miejsc i ich rezerwacja. Przypominamy również, że aplikacja rezerwacji biletów używa *ConversationScope* do śledzenia poszczególnych użytkowników. Warto więc uwzględnić ten element w testach.

Na szczęście, kontener weld zapewnia wszystko, czego potrzebujemy: *org.jboss.weld.context.bound.BoundConversationContext* musi zostać wstrzyknięty do klasy testu.

```
@Inject BoundConversationContext conversationContext;

@Before
public void init() {
    conversationContext.associate(
        new MutableBoundRequest(new HashMap<String, Object>(),
            new HashMap<String, Object>()));
    conversationContext.activate();
}
```

Adnotacja `@Before` powoduje, że kod zostanie wykonany przed poszczególnymi metodami testowymi, ale po wstrzyknięciach. W przedstawionej sytuacji służy do powiązania `conversationContext` z `MutableBoundRequest` przed aktywacją za pomocą `conversationContext.activate`. To niezbędne, by uzyskać symulację konwersacji z poziomu środowiska testowego Arquillian.

Warto wyjaśnić, że interfejsy `BoundRequest` są zdefiniowane w API `weird` i mają za zadanie przechowywać konwersację obejmującą kilka żądań, ale krótszą niż sesja.

Poniżej znajduje się pełna wersja klasy `TicketTest` zawierająca w metodzie `testTicketAgency` dodatkowo tworzenie obiektu `Theatre` i rezerwację miejsca.

```
@RunWith(Arquillian.class)
public class TicketTest {

    @Inject BoundConversationContext conversationContext;

    @Before
    public void init() {
        conversationContext.associate(
            new MutableBoundRequest(new HashMap<String, Object>(),
                new HashMap<String, Object>()));
        conversationContext.activate();
    }

    @Deployment
    public static Archive<?> createTestArchive() {
        return ShrinkWrap.create(WebArchive.class, "ticket.war")
            .addPackage(SeatProducer.class.getPackage())
            .addPackage(Seat.class.getPackage())
            .addPackage(TicketService.class.getPackage())
            .addPackage(DataManager.class.getPackage())
            .addAsResource("META-INF/persistence.xml")
            .addAsWebInfResource(EmptyAsset.INSTANCE, "beans.xml");
    }

    @Inject
    TicketService ticketService;

    @Inject
    BookerService bookerService;

    @Inject
    Logger log;

    @Test
    public void testTicketAgency () throws Exception {

        SeatType seatType = new SeatType();
```

```

seatType.setDescription("Balkon");
seatType.setPrice(20);
seatType.setQuantity(5);

ticketService.createSeatType(seatType);
log.info("Utworzono typ miejsc.");
assertNotNull(seatType.getId());

List<SeatType> listSeats = new ArrayList();
listSeats.add(seatType);
ticketService.createTheatre(listSeats);

log.info("Utworzono teatr");
log.info(seatType.getDescription() + " zostało utrwalone
↳ z identyfikatorem " + seatType.getId());

bookerService.bookSeat(new Long(seatType.getId()), seatType.getPrice());
log.info("Dokonano rezerwacji");
log.info("Pozostała kwota: " +bookerService.getMoney());
assertTrue(bookerService.getMoney() <100);
}
}

```

Wynik działania metod logowania w konsoli serwera aplikacji powinien być podobny do poniższego.

```

E:\app\jboss\bin\standalone.bat
C:\> "E:\app\jboss\...
name: primary
12:17:19,998 INFO [org.hibernate.service.jdbc.connections.internal.ConnectionProviderInitiator] (MSC service thread 1-5) HH000130: Instantiat
vider: org.hibernate.eib.connection.InjectedDataSourceConnectionProvider
12:17:20,024 INFO [org.hibernate.dialect.Dialect] (MSC service thread 1-5) HH000400: Using dialect: org.hibernate.dialect.MySQLDialect
12:17:20,025 INFO [org.hibernate.engine.transaction.internal.TransactionFactoryInitiator] (MSC service thread 1-5) HH000268: Transaction strat
TransactionInternal
jta.CMTTransactionFactory
12:17:20,028 INFO [org.hibernate.hql.internal.ast.ASTQueryTranslatorFactory] (MSC service thread 1-5) HH000397: Using ASTQueryTranslatorFactor
12:17:20,052 INFO [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-5) HH000227: Running hbm2ddl schema export
12:17:20,055 ERROR [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-5) HH000389: Unsuccessful: alter table Seat drop foreign key
12:17:20,056 ERROR [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-5) Table 'ticketsystem.seat' doesn't exist
12:17:20,093 INFO [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-5) HH000230: Schema export complete
12:17:20,096 INFO [org.jboss.weld.deployer] (MSC service thread 1-3) JBAS016008: Starting weld service for deployment ticket-agency-test.war
12:17:20,197 INFO [org.jboss.web] (MSC service thread 1-6) JBAS018210: Registering web context: /ticket-agency-test
12:17:20,215 INFO [org.jboss.as.connector.management.handler.thread-6] JBAS018558: Deployed "ticket-agency-test.war"
12:17:20,558 INFO [com.pactpub.as7development.chapter6.service.TicketService] (http-127.0.0.1-8080-1) Rejestracja Balkon
12:17:20,570 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Utworzono typ miejsc Balkon
12:17:20,593 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Utworzono teatr
12:17:20,594 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Balkon zostało utrwalone z identyfikatorem 1
12:17:20,596 INFO [com.pactpub.as7development.chapter6.service.BookerService] (http-127.0.0.1-8080-1) Rezerwuj miejsce 1
12:17:20,632 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Miejsce zarezerwowane.
12:17:20,633 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Dokonano rezerwacji
12:17:20,634 INFO [com.pactpub.as7development.chapter6.test.TicketTest] (http-127.0.0.1-8080-1) Pozostała kwota: 89
12:17:20,660 INFO [org.jboss.weld.deployer] (MSC service thread 1-7) JBAS016009: Stopping weld service for deployment ticket-agency-test.war
12:17:20,663 INFO [org.jboss.as.jpa] (MSC service thread 1-2) JBAS011409: Stopping Persistence Unit service 'ticket-agency-test.war@primary'
12:17:20,664 INFO [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-2) HH000227: Running hbm2ddl schema export
12:17:20,690 INFO [org.hibernate.tool.hbm2ddl.SchemaExport] (MSC service thread 1-2) HH000230: Schema export complete
12:17:20,714 INFO [org.jboss.as.server.deployment] (MSC service thread 1-8) JBAS015877: Stopped deployment ticket-agency-test.war in 58ms
12:17:20,727 INFO [org.jboss.as.repository] (management-handler-thread - 10) JBAS014901: Content removed from location E:\app\jboss\standalone\
84da6e8f6f8c13d1eba5a12727b\content
12:17:20,729 INFO [org.jboss.as.server] (management-handler-thread - 10) JBAS018558: Undeployed "ticket-agency-test.war"
java.exe:9384
130421[64] 1/1 (+) CAPS NUM SCR: PRI: (0,443):(147,482) 148x40 148x2000 0 4 74 25V 1432/7540 100%

```

Informacje dodatkowe

Projekt Arquillian cały czas ewoluuje. Opis wszystkich jego możliwości wykracza poza ramy tej książki. Więcej informacji na jego temat można znaleźć w dokumentacji dostępnej pod adresem <http://arquillian.org/guides/>.

Podsumowanie

W tym rozdziale przyjrzelśmy się dokładniej jednemu z najistotniejszych elementów systemów biznesowych — testom integracyjnym. Historycznie jednym z najtrudniejszych elementów Javy EE były właśnie testy, ale projekt Arquillian znakomicie to rozwiązuje.

Stosowany jako rozszerzenie frameworka JUnit Arquillian zapewnia doskonałe warunki do tworzenia testów integracyjnych logiki biznesowej aplikacji Javy EE.

Arquillian podłącza się pod tryb życia frameworka testów, zapewniając odpowiednią reakcję na zdarzenia i zarządzanie kontenerem (jego uruchamianie i zatrzymywanie). Dodatkowo tworzy specjalne archiwum z możliwością wdrożenia, które zawiera klasy testów oraz wszystkie niezbędne zależności i zasoby.

W następnym rozdziale omówimy tworzenie aplikacji przy użyciu dostawcy komunikatów JBoss (HornetQ), wprowadzając przykłady wykorzystania ziaren sterowanych komunikatami.

Skorowidz

./jboss-cli.sh, 196
?stateful, 65
@ActivationConfigProperty, 164
@ApplicationPath, 187
@ApplicationScoped, 87
@Asynchronous, 78
@Before, 146
@Clustered, 221, 222, 224
@ConversationScoped, 87
@Dependent, 87
@Deployment, 141
@EJB TheatreBox, 59
@ElementCollection, 19
@Entity, 110, 118
@GeneratedValue, 118
@GET, 186
@Id, 110, 118
@Inject, 92, 93, 95
@javax.annotation.PostConstruct, 51
@javax.annotation.Resource, 226
@javax.annotation.security.
 DenyAll, 249
@javax.annotation.security.
 PermitAll, 249
@javax.annotation.security.
 RolesAllowed, 249
@javax.annotation.security.RunAs, 249
@javax.ejb.
 ConcurrencyManagement, 58
@javax.ejb.Singleton, 51
@javax.ejb.Startup, 51
@javax.jws.SOAPBinding, 176
@javax.persistence.Entity, 110
@javax.persistence.Id, 110

@javax.validation.constraints.
 NotNull, 119
@javax.validation.constraints.
 Pattern, 119
@javax.validation.constraints.
 Size, 119
@JoinColumn, 118
@Lock, 57
@ManyToOne, 118
@MessageDriven, 159
@Model, 93, 94, 126
@Named, 85, 92, 98, 122, 126,
 127
@NotNull, 119
@OneToMany, 118
@OrderColumn, 19
@org.jboss.ejb3.annotation.
 Clustered, 220, 221, 224
@org.jboss.ejb3.annotation.
 SecurityDomain, 249, 252
@org.jboss.ws.api.annotation.
 Webcontext, 252
@Path, 186
@POST, 186
@PostConstruct, 60, 226
@Producer, 94
@Produces, 98
@Remote(TheatreInfo.class), 59
@RequestScoped, 84, 87
@Resource, 75, 121, 226
 lookup, 226
@ResourceAdapter, 168
@RolesAllowed, 250, 262
@RunWith, 141
@Schedule, 76
@SecurityDomain, 250, 252
@SessionScoped, 87, 92, 236

@SessionScoped, 235
@Stateful, 60, 222
@Stateless, 59, 221
@Table, 118
@Timeout, 75
@WebContext, 252
@WebMethod, 177
@WebParam, 177
@WebResult, 177
@WebService, 176, 251
@WebServlet, 41

A

a4j:poll, 104
activeCount, 212
ActiveMQ, 165, 166
 instalacja, 166
 integracja z HornetQ, 165
 konfiguracja serwera, 166
 konsumpcja komunikatów,
 168
 opis atrybutów, 167
wdrożenie, 168
activemq-rar-5.7.0.txt, 166
adapter JCA, 165
adapter zasobów, 163
 ActiveMQ, 166
 Javy, 165
addPackage, 141
add-user.bat, 36, 74, 217, 242
add-user.sh, 36, 74, 217, 242
admin, 247, 248
adres zasobu, 198
advertise-socket, 233
Adwords, 18

- affinity, 222
- akcje CRUD, 186
- algorytm
 - MD5, 248
 - RSA, 256
 - SHA, 248
- all-relevant-server-groups, 48
- all-server-groups, 47, 202
- Apache CXF, 173
 - klasy pomocnicze, 182
 - kod klienta, 183
 - mechanizmy dziennika zdarzeń, 183
 - warstwa integracji, 173
- API
 - bezpieczeństwa, 239
 - Javy EE, 240
 - bezpośrednie, 209
 - CDI dla Javy, 90, 109
 - Common Annotations, 90
 - EJB, 90
 - EJB 3.1, 55
 - encji, 119
 - getCallerPrincipal(), 241
 - getUserPrincipal(), 241
 - HttpClient, 190
 - Java Persistence, 134
 - java.util.logging, 116
 - Javy dla usług sieciowych, 20
 - bazujących na XML, 20
 - jawne wyłączenie części, 116
 - jboss-ejb-api, 67
 - JPA, 110
 - klienta RESTEasy, 192
 - logów JBoss, 90
 - org.jboss.marshalling, 67
 - org.jboss.xnio, 67
 - RichFaces, 103
 - SecurityDomain, 250
 - slf4j, 116
 - tożsamości użytkownika, 241
 - transakcyjnego JBoss, 67
- API zarządzania
 - bezpośrednie, 209
 - odczytywanie opisów modelu zarządzania, 209
 - beztypowe, 209
 - jboss-as-controller-client, 209
 - jboss-dmr, 209
 - obiekt żądania operacji, 210
 - użycie do sterowania serwerem aplikacji, 209
- aplikacja
 - bezpieczeństwo, 239
 - Database, 249
 - moduł logowania, 243
 - EJB
 - adnotacje bezpieczeństwa, 249
 - dodanie ziarna
 - bezzastanowowego, 59
 - dodanie ziarna sesyjnego, 60
 - konfiguracja Maven, 61
 - konfiguracja pliku pom.xml, 54, 67
 - lista dowiązań JNDI, 65
 - Maven, 51
 - miejsce zabezpieczeń, 255
 - ograniczenia bezpieczeństwa, 250
 - powiązania elementów konfiguracji, 251
 - przygotowanie ziaren sesyjnych, 58
 - rola bezpieczeństwa, 250
 - sterowanie współbieżnością ziarna, 57
 - tworzenie kodu, 55
 - wdrożenie, 61
 - wdrożenie z poziomu środowiska Eclipse, 62
 - wyjątek sprawdzany, 61
 - wyjątek wykonania, 61
 - zgłaszane wyjątki, 60
 - ziarno singletonowe, 57
 - zwiększona przenośność, 59
- Java EE 6, 109
- JBoss Forge, 268
 - budowanie i wdrożenie, 271
 - demo, 272
 - interfejs graficzny, 270
- JMS
 - dodanie ziaren sterowanych komunikatami, 156
 - elementy, 151
 - generowanie wiadomości, 159
 - informowanie o zmianach, 160
 - komunikat po rejestracji użytkownika, 160
- optymalizacja połączeń
 - JMS, 162
 - wdrożenie, 161
- JSF
 - wersja klastrowa, 236
 - wysoka dostępność, 235
- klastrowa, 223
 - wdrażanie, 220
 - wymuszenie użycia bibliotek, 220
- MVC, 136
- testowanie, 135
- trwałego magazynu danych, 109
 - dodanie konfiguracji Maven, 115
 - dodanie kontrolera sterującego zadaniami użytkowników, 126
 - dodanie trwałości, 111
 - dodanie usług, 123
 - dodanie walidacji ziarna, 118
 - instalacja sterownika JDBC, 112
 - klasy produkujące, 120
 - kod zapytań, 123
 - konfiguracja bazy danych, 111
 - konfiguracja trwałości, 119
 - projekt Maven, 114
 - tworzenie encji, 116
 - uruchomienie przykładu, 132
 - utworzenie nowego źródła danych, 113
 - warstwa sesyjna, 125
 - widoki JSF, 128
- typu enterprise, 51
 - komponenty EJB, 49
- usługi sieciowe, 171
 - dodanie obsługi REST, 187
 - dodanie usługi SOAP, 172
 - testy, 185
- usuwanie wdrożenia, 201, 202, 203
- wdrażanie przy użyciu CLI, 201
 - do domeny JBoss AS 7, 202
 - do kilku węzłów JBoss AS 7, 203

- do pojedynczej grupy serwerów, 202
- na wszystkich grupach serwerów, 202
- wdrożenie na serwerze, 39
 - dodanie do listy wdrożonych zasobów, 41
 - dodanie klasy, 40
 - lokalizacja, 39
 - pełna publikacja, 42
 - przy użyciu narzędzia CLI, 46
 - wskazanie korzenia kontekstu, 41
 - z poziomu konsoli
 - webowej, 43
 - zaawansowane opcje, 42
- webowa, 88
 - dodanie strony początkowej, 100
 - dołączenie bibliotek, 103
 - dołączenie
 - harmonogramów, 102
 - dołączenie zewnętrznego systemu, 102
 - elementy strony głównej, 98
 - główna struktura, 96
 - główny panel aplikacji, 96
 - instalacja RichFaces, 103
 - kod ziaren, 91
 - miejsce zabezpieczeń, 255
 - mod_cluster, 232
 - mod_jk, 232
 - mod_proxy, 232
 - niezbędne zależności, 90
 - obserwacja obiektów, 94
 - obsługa JSF 2, 99
 - przygotowanie do uruchomienia, 100
 - równoważenie obciążenia, 232
 - stopka, 96
 - strona szablonu, 96
 - struktura projektu, 101
 - tworzenie klastrów, 231
 - tworzenie widoku, 95
 - uruchomienie, 106
 - wersja rozbudowana, 103
 - widok strony głównej, 97
 - widok w przeglądarce, 101
 - zgodna z klastrami, 235
 - zmiana na liście miejsc, 94
- wykorzystująca JBoss JMS Provider, 149
- appclient, 34
- ApplicationRealm, 74
- application-roles.properties, 242
 - role użytkowników, 243
- application-users.properties, 242
 - nazwa użytkownika, 243
- app-name, 64
- archetypy
 - dotyczące szybkiego startu, 71
 - EJB, 52
 - ekran szczegółów, 66
 - jboss-javaee6-webapp-archetype, 114
 - jboss-javaee6-webapp-ear-archetype, 114
 - Maven, 88
 - maven-archetype-quickstart, 65
 - webapp-javaee6, 88, 114, 175
- Architektura Javy dla dowiązań XML, 20
- arq-jbossas-managed, 140
- arq-jbossas-remote, 140, 143, 144
- Arquillian, 134, 135, 136
 - adres i port systemu zarządzania, 142
 - dodanie konfiguracji, 142
 - działanie, 136, 148
 - informacje dodatkowe, 148
 - konfiguracja pliku pom.xml, 139
 - kontener
 - weld, 140
 - zdalny, 144
- korzystanie z narzędzia, 137
- lokalizacja serwera JBoss AS 7, 145
- pierwszy test, 141
- pisanie testu, 137
- profile, 140
- protokół
 - do komunikacji z serwerem, 140
- Servlet 3.0, 142
- struktura folderów, 137
- stymulacja konwersacji z poziomu środowiska testowego, 146
- wynik testu, 144
- arquillian.xml, 142, 145
- artefakty
 - forge-demo.var, 271
 - hibernate-validator, 115, 116
 - jboss-as-maven-plugin, 62
 - jboss-javaee-6.0-with-hibernate, 115
 - jboss-javaee-6.0-with-tools, 115
 - maven-compiler-plugin, 62
 - maven-ejb-plugin, 62
 - ticket-agency-ejb, 67
- as7 deploy, 271
- as7 undeploy, 272
- asInt, 209
- asynchroniczne komponenty EJB, 50
- atrybuty
 - ?stateful, 65
 - activeCount, 212
 - Auto-deploy Exploded, 46
 - Auto-deploy Zipped, 46
 - Deployment timeout, 46
 - Enabled Path Relative to, 46
 - finalName, 62
 - hibernate.hbm2ddl.auto, 120
 - hibernate.show_sql, 120
 - name, 45, 120, 176
 - Path, 45
 - Scan Interval, 46
 - SeatType, 142
 - skanera, 45
 - value, 98
- Auto-deploy Exploded, 46
- Auto-deploy Zipped, 46
- automaticCustomer, 77
- automatyczne
 - generowanie encji, 116
 - uzupełnianie, 200
- autoryzacja, 240
 - metody EJB, 249
- auto-start, 220

B

- b, 215
- Base64, 218, 243, 248
- Base64Utils, 248
- BASIC, 243
- baza danych
 - automatyczne generowanie encji, 116
 - konfiguracja, 111
 - mysql, 111
 - MySQL, 111
 - tabela, 111
 - jeden do n, 111
 - klucz obcy, 111
 - schemat, 111
- Bean Pools, 158
- bean-name, 64
- beans.xml, 88, 100
 - schemat XML, 88
- begin(), 125
- bezinterfejsowy komponent EJB, 50
- bezpieczeństwo aplikacji, 239
 - adnotacje, 249
 - API bezpieczeństwa w języku Java, 239
 - autoryzacja, 240
 - bazujące na formularzu, 245
 - certyfikat, 254
 - deklaratywne, 240
 - deskryptor wdrożenia webowego, 244
 - domena bezpieczeństwa, 242
 - formularz logowania, 249
 - generowanie certyfikatów, 259
 - kontenery komponentów, 240
 - kryptografia asymetryczna, 253
 - miejsca zabezpieczeń, 255
 - moduł logowania
 - pierwszy, 242
 - użycie, 243
 - wykorzystujący bazę danych, 246
 - narzędzia do zarządzania certyfikatami, 255
 - ograniczenia bezpieczeństwa, 244

- opcje zarządzania, 241
- podsystem bezpieczeństwa JBoss AS 7, 241
- programowe, 240
- protokół szyfrowania informacji, 252
- szyfrowanie symetryczne, 253
- uruchamianie komunikacji SSL, 255
- uwierzytelnienie, 240
- zabezpieczenie
 - hasel, 247
 - interakcji ze stronami WWW, 254
 - komponentów EJB, 249
 - komunikacji EJB, 258
 - komunikacji HTTP, 256, 257
 - warstwy transportowej, 252
 - zasada bezpieczeństwa, 260
 - użycie, 261
- bezpieczeństwo usług sieciowych, 251
- bezpośrednie API zarządzania, 209
- bezstanowe ziarna sesyjne, 49
- bezstanowy EJB, 223
- beztypowe API zarządzania, 209
 - narzędzie stałej obserwacji zasobów, 210
 - użycie, 209
- biblioteka
 - JAXB, 174
 - JGroups, 214
 - JSON.simple, 191
 - mysql-connector-java-5.X.XX-bin.jar, 112
 - Netty, 152
 - obiektów naśladowujących, 136
 - RESTEasy, 187
 - RichFaces, 102
- Bill of Materials, 54, 91, 103, 115
- bin, 34
- bin/client, 34
- body, 172
- BOM, 54, 91, 103, 115
 - junit, 139
- book.xhtml, 128, 130

- BookerService, 123, 125, 160
 - price, 169
 - seatId, 169
- bookSeat, 99, 125, 182
 - zabezpieczenie, 250
- bookSeatAsync, 78, 79
- BoundRequest, 146
- broker
 - ActiveMQ, 168
 - komunikatów Apache, 165
- Brontes, 22
- Browse workspace, 62
- build, 271
- buyTicket, 57, 58, 77, 189, 191

C

- c, 215
- CA, 254
- CacheContainer, 226
- CacheLoader, 225
- CacheManager, 225
- calculatePower, 175
- cancelTimers, 77
- CASE, 19
- cat, 269
- cd, 199
- CDI, 13, 17, 20, 83, 84, 107
 - implementacja w JBoss, 87
 - łączenie trwałości, 109
 - podstawowy element, 84
 - przekształcenie systemu rezerwacji biletów, 88
 - tworzenie kodu ziaren, 91
 - wprowadzenie, 83
 - zakresy, 86
 - ziarna nazwane, 85
- cele JMS, 151, 155
 - dodawanie skryptu CLI, 205
 - konfiguracja, 155
 - określenie filtru, 156
 - ticketQueue, 159
- Certificate Signing Request, 257
- Certification Authority, 254
- certreq.csr, 257
- certyfikat
 - cyfrowy, 254
 - klienta generowanie, 259

- klucza publicznego, 254
 - narzędzia do zarządzania, 255
 - podpisanie, 257
 - podpisany przez CA
 - zabezpieczenie komunikacji HTTP, 257
 - podpisany przez samego siebie, 256
 - zabezpieczenie komunikacji HTTP, 256
 - serwera
 - generowanie, 259
 - zaimportowanie, 259
 - zaufany po stronie klienta, 259
 - CLI, 21, 24, 46, 195
 - automatyczne uzupełnianie, 47, 48
 - fragmentów ścieżek, 201
 - dodawanie zasobów JMS, 205
 - domyślny folder, 201
 - instalacja źródła danych jako modułu, 204
 - instrukcje warunkowe, 204
 - konstrukcja poleceń, 197
 - automatyczne uzupełnianie, 200
 - określanie adresu zasobu, 198
 - opracje na zasobach, 198
 - lista
 - aktualnie wdrożonych aplikacji, 201
 - dostępnych typów węzłów, 200
 - nazw dostępnych dla danego typu węzła, 200
 - monitorowanie zasobów serwera, 210
 - operacje dostępne dla wybranego węzła, 200
 - parametry operacji, 200
 - restart węzłów aplikacji serwerowej, 204
 - tryb graficzny, 197
 - tworzenie skryptów, 203
 - użycie zaawansowanych języków, 205
 - użycie języków skryptowych do wykonywania operacji, 206
 - wdrażanie aplikacji, 46, 201
 - do kilku węzłów JBoss AS 7, 203
 - domyślne wdrożenie, 47
 - na jednej grupie serwerów, 202
 - na wszystkich grupach serwerów, 202
 - ponowne wdrożenie, 47
 - w domenie, 47
 - wskazywanie lokalizacji, 47
 - wykonywanie poleceń z poziomu wielu języków, 206
 - wykorzystanie, 199
 - wyświetlanie poleceń, 46
 - zatrzymanie serwera, 25
 - żądanie wykonania operacji, 197
 - client.createRequest, 190
 - client.get, 190
 - clientPublicKey.cer, 259
 - ClientRequest, 190
 - elementy adresu, 190
 - ClientRequestFactory, 190
 - COALESCE, 19
 - com.mysql, 112
 - Command Line Interface, 24, 46
 - commandButton, 99
 - commit(), 125
 - Common Annotations API, 55
 - Community edition, 28
 - Company, 110
 - composition, 96
 - ConcurrencyManagementType.BEAN, 58
 - configuration, 34, 35
 - connect, 25, 196, 207
 - connection-url, 113
 - content, 35, 96
 - Context and Dependency Injection, 13, 17, 20, 83
 - ConversationScope
 - w testach, 145
 - create, 141, 184
 - Create, Read, Update, Delete, 186
 - createSeatType, 125, 142
 - createTheatre, 125
 - CRON, 76
 - CRUD, 186, 270
 - odzworowanie akcji na HTTP, 186
 - customNamedBean, 86
 - czasomierz, 75
 - automatyczny, 75
 - nietrwały, 77
 - programowy, 75
 - tworzenie, 75
 - trwały, 77
- ## D
- dane
 - trwałość i standardy, 109
 - użytkowników, 36
 - DAO, 136
 - data, 35, 178
 - Data Access Object, 136
 - Database, 242, 246, 249
 - DataManager, 122, 123
 - dataTable, 98, 130
 - value, 98
 - default, 216
 - default.xhtml, 96, 131
 - defaultProtocol, 142
 - definicje instancji fabryki połączeń, 153
 - DELETE, 186
 - deleteAllData, 123
 - demouser, 242
 - Denial Of Service, 107
 - deploy, 46, 47, 199, 201
 - all-server-groups, 202
 - f, 201
 - projects as compressed archives, 43
 - deployment, 42, 201
 - Deployment Scanners, 45
 - Deployment timeout, 46
 - deployments, 35, 166
 - description, 119
 - deskryptor
 - wdrożenia EJB, 240
 - JCA, 166
 - deszyfracja, 252
 - disabled, 47
 - distinct-name, 64
 - distributable, 236
 - Djava.endorsed.dirs, 226

doCleanup, 125
 docs/example, 34
 docs/schema, 34
 DOCUMENT, 176
 dodawanie
 encji, 116
 użytkowników, 36
 źródła danych, 205
 dokument WSDL, 172
 domain, 34, 229
 struktura zawartości, 35
 domain.xml, 113, 216
 full-ha, 216
 mod_cluster, 233
 domena
 bezpieczeństwa, 242
 dla komponentów EJB, 262
 dodanie referencji, 250
 klasy lub metody, 249
 moduł logowania
 wykorzystujący
 bazę danych, 246
 mysqldomain, 252
 określanie, 242
 przykład definicji, 260
 usługi sieciowe bazujące
 na EJB, 251
 elementy, 31
 kontroler domeny, 31
 kontroler hosta, 32
 węzły serwerów
 aplikacyjnych, 32
 domyślny mechanizm
 bezpieczeństwa, 36
 dostarczanie komunikatów, 151
 dostęp do danych, 136
 dowiązania JNDI, 64
 dla komponentu EJB, 65
 dziennik serwera, 65
 JBoss, 65
 utworzenie tekstu
 wyszukiwania
 w kliencie EJB, 65
 driver, 113
 Durable, 156
 duration, 75
 Dynamic web module version, 40
 Dynamic Web Project, 51
 dziennik zdarzeń
 interceptory, 183
 klastry, 220

E

EAI, 165
 EAP, 22
 EAR, 19
 Eclipse, 26, 114
 Browse workspace, 62
 dodanie zależności do
 projektu, 115
 instalacja, 26
 dodatku JBoss AS, 27
 narzędzie Maven, 51
 pisanie testu Arquillian, 137
 Project Properties, 117
 Run Configuration, 62
 soapUI, 179
 wdrożanie aplikacji, 62
 pierwszej aplikacji, 39
 Window/ Show View/JUnit,
 144
 zaawansowane opcje
 wdrożenia, 42
 zmiana ustawień
 domyślnych, 42
 zdalny klient EJB, 73
 EJB, 17, 20, 49, 107, 109
 bezzastanowe, 223
 deskryptor wdrożenia, 240
 getCallerPrincipal(), 241
 isCallerInRole(), 240
 model bezpieczeństwa, 239
 monitorowanie zasobów
 serwera, 210
 rodzaje komponentów, 49
 singleton, 223, 224
 stanowe, 223
 zabezpieczanie komunikacji,
 258
 zdalny klient, 227
 EJB 3.1
 metody asynchroniczne, 77
 nowe funkcjonalności, 49
 usprawnienia, 19
 ejb-client, 67
 ejbclientalias, 259
 EJBException, 61
 ejb-jar.xml, 240, 250, 252
 ejb-javaee6, 53
 EJBRealm, 260
 ejb-roles.properties, 260

ejb-security-domain, 260
 ejb-users.properties, 260
 EL, 85
 e-mail, 79
 Enabled Path Relative to, 46
 encja
 dodawanie, 116
 EJB, 109
 generowanie ze schematu
 bazy danych, 116
 JPA, 110
 producenty, 121
 Seat, 118
 SeatType, 119
 tworzenie, 116
 endpoint.name, 71
 Enterprise Application
 Integration, 165
 Enterprise ARchive, 19
 Enterprise Java Persistence, 109
 Enterprise JavaBeans, 17, 19, 49,
 109
 bezpieczeństwo
 deklaratywne, 240
 Enterprise Platform, 22
 Entity, 269
 EntityManager, 121
 envelope, 172
 error.jsf, 245
 exec-maven-plugin, 72
 execute, 210
 Executor, 79

F

-f, 47
 fabryka połączeń, 151
 ActiveMQ, 167
 definicje instancji, 153
 InVmConnectionFactory, 153
 optymalizacja połączeń, 162
 RemoteConnectionFactory,
 153
 rodzaje, 153
 stosująca pulę połączeń
 wewnątrz producentów
 komunikatów, 163
 tworzenie, 153
 wstrzykiwanie, 154
 wykorzystanie, 153

wykorzystująca pulę połączeń, 154
zmiana ustawienia dowiązań JNDI, 154

facelety, 19, 95
budowanie, 96

FacesContext, 121

FacesMessages, 93

farm, 203
--file, 203

finalName, 62

findSeat, 77

finish, 130

foldery
appclient, 34
bin, 34, 111
bin/client, 34
configuration, 34, 35
content, 35
data, 35, 178
deployment, 201
deployments, 35, 166
doc/as, 166
docs/example, 34
docs/schema, 34
domain, 34
farm, 203
główny folder serwera, 34
HelloWorld.war, 42
java, 137
JBOSS_HOME/bin, 201
JYTHON_HOME/bin, 207
lib/ext, 35
log, 35
main, 112, 137
META-INF, 88, 167, 240
modules, 34, 233
resources, 137
servers, 35
standalone, 34
target, 271
test, 137
test/java, 137
tmp, 35
views, 128
webapp, 90, 97
WEB-INF, 88, 100, 240
welcome-content, 34

footer, 96

forge install-plugin jboss-as-7, 271

forge.bat, 266

forge.sh, 266

FORGE_HOME, 265

forge-demo.war, 271

format x.509, 254

formularz
logowania, 245
pola formularza, 245
wprowadzanie nowego rodzaju miejsc, 130

framework
Arquillian, 134
Hibernate, 110, 112
kliencki JAX-RS, 187
ORM, 110
wstrzykiwania zależności, 121

full, 153, 216

Full Publish, 42

full-ha, 153, 216

fully-qualified-classname-of-the-remote-interface, 64

funkcja automatycznego uzupełniania, 47, 48

funkcje
Apache CXF, 173
isCallerInRole(), 240
isUserInRole(), 240
skrótów, 248

funkcjonalności JAX-WS, 173

Future, 78
kod korzystający z obiektu, 79

futureResult, 80

G

generate-entities, 269

generowanie certyfikatów, 259

get, 226

GET, 186

getCallerPrincipal(), 241

getSeatList, 57, 189

getSeatPrice, 57

getSeats, 94, 182

getType(), 209

getUser, 187

getUserPrincipal(), 241

główny folder serwera, 34

Google Guice, 152

Google trends, 18

Google Web Toolkit, 36

grid, 104

grupa serwerów, 33

gui, 197

GWT, 36

H

ha, 216

handleGETRequest, 186

handlePOSTRequest, 187

harmonogram wywołań metod zwrotnych, 76

hasła
aktualizacja, 248
algorytm MD5, 248
Base64, 248
szyfrowanie, 247
wstawianie do bazy danych, 248

header, 172

heap-memory-usage, 208

HelloWorld.war.dodeploy, 42

HelloWorld.war.failed, 42

help, 267

Hibernate, 19, 110
Persistence API, 115
procesor adnotacji, 116
sterownik JDBC, 112

hibernate.hbm2ddl.auto, 120

hibernate.show_sql, 120

hibernate-tools, 268

hibernate-validator, 115

HornetQ, 152
adapter zasobów, 163
integracja z ActiveMQ, 165
współpraca z adapterem JCA, 152
zintegrowany z JBoss AS jako moduł, 152

host
auto - start, 220
dodanie hasła, 219
host.xml, 218
-host-config, 218
jboss.domain.master.address, 218
konfiguracja, 218
kontroler, 218
nazwa, 218
port-offset, 220
uruchamianie, 220

- host.xml, 216
 - konfiguracja, 218
 - węzłów serwera, 219
 - referencja do lokalnego kontrolera domeny, 217
 - host-config, 218
 - HTTP, 149, 249
 - zabezpieczenie komunikacji certyfikatem
 - podpisanym przez CA, 257
 - podpisanym przez samego siebie, 256
 - HTTP Digest, 36
 - HTTP load balancer, 231
 - httpd.conf, 234
 - Hypertext Transfer Protocol, 149
- I**
- IDE Eclipse
 - pobieranie, 26
 - implementacja
 - CDI w JBoss AS, 87
 - index.xhtml, 97, 100
 - przestrzenie nazw RichFaces, 103
 - Infinispan, 215, 224, 225, 229
 - InfoBean
 - bookSeat, 182
 - inspekcja adaptera JMS, 166
 - instalacja
 - adaptera zasobów ActiveMQ, 166
 - Forge, 265
 - Javy SE, 22
 - JBoss AS 7, 22
 - mod_cluster, 233
 - narzędzia Maven, 29
 - narzędzi JBoss, 27
 - RichFaces, 103
 - soapUI, 179
 - środowiska Eclipse, 26
 - integracja
 - aplikacji biznesowych, 165
 - HornetQ i ActiveMQ, 165
 - JMS i JCA
 - a usługi sieciowe, 165
 - obsługa adaptacji zasobów, 165
 - IntelliJ IDEA, 28
 - interceptory dziennika zdarzeń, 183
 - interfejs
 - BoundRequest, 146
 - CacheLoader, 225
 - CLI, 24
 - Java Persistence API, 110
 - javax.cache.Cache, 225
 - javax.jms.MessageListener, 151
 - końcówki usługi sieciowej, 180
 - logowania, 245
 - QueueReceiver, 152
 - QueueSender, 152
 - TimerService, 75
 - TopicPublisher, 152
 - TopicSubscriber, 152
 - usługi sieciowej, 181
 - webowy, 36
 - wiersza poleceń, 26, 46, 113, 195
 - zarządzania, 196, 218
 - z poziomu wiersza poleceń, 21
 - InVmConnectionFactory, 153
 - IronJacamar, 166
 - IronJacamar 1.1, 166
 - ironjacamar.xml, 166
 - isCallerInRole(), 240
 - isDone, 80
 - isUserInRole(), 240
- J**
- j_password, 245
 - j_security_check, 245
 - j_username, 245
 - J2EE, 17
 - JAAS, 263
 - Jakarta Commons, 190
 - java, 137
 - java:/jaas/, 242
 - java:/jaas/other, 244
 - java:/jmsXA, 154, 162
 - java:app, 64
 - java:global, 64
 - java:jboss/
 - jms/queue/ticketQueue, 159
 - java:module, 64
 - Java API for RESTful Web Services, 17
 - Java API for XML Web Services, 17
 - Java Architecture for XML Binding, 174
 - Java Authentication and Authorization Service, 263
 - Java Beans
 - komponenty, 118
 - Java Connector Architecture, 165
 - Java EE, 17
 - usługi bezpieczeństwa, 239
 - Java EE 6
 - API Javy dla usług sieciowych, 20
 - bazujących na XML, 20
 - Architektura Javy dla dowiązań XML, 20
 - CDI, 83
 - Contexts and Dependency Injection for Java, 20
 - Enterprise JavaBeans, 19
 - Java Persistence API, 19
 - Java Servlet API 3.0, 20
 - JavaServer Faces, 18
 - JAX-RS, 20
 - komponenty EJB, 49
 - nowe usprawnienia, 18
 - zarządzanie komponentami, 84
 - Java Enterprise Edition, 49
 - Java Message Service, 14, 149
 - Java Messaging Service, 17
 - Java Naming Directory Index, 153
 - Java Persistence API, 17, 19, 109
 - Java Persistence Query Language, 110
 - Java SE
 - instalacja, 22
 - test, 22
 - Java Secure Socket Extension, 255
 - Java Server Faces, 17, 83
 - Java Server Pages, 17
 - Java Servlet API 3.0, 20
 - Java Transaction API, 139
 - java -version, 22
 - java.ejb.Schedule, 75
 - java.ejb.Schedules, 75
 - java.util.concurrent.Future, 78

- java.util.CurrentMap, 225
- java.util.Logger, 121, 159
- JavaBeans Validation, 118
- JavaServer Faces, 18
- javax.activation, 87
- javax.annotation, 87
- javax.annotation.security.
 - RolesAllowed, 250
- javax.cache.Cache, 225
- javax.ejb, 87
- javax.ejb.LockType.READ, 57
- javax.ejb.LockType.WRITE, 57
- javax.ejb.Singleton, 50
- javax.enterprise.context.
 - Conversation, 87
- javax.enterprise.event.Event, 94
- javax.jms, 87
- javax.jms.MessageListener, 151
- javax.persistence, 87
- javax.security, 87
- javax.servlet.http.HttpSession, 235
- javax.transaction, 87
- javax.xml, 87
- jawne wyłączenie części API, 116
- JAXB, 174
- JAXB 2.0 XML to Java Mapping, 174
- JAXB 2.2, 20
- JAX-RS, 17, 20, 171, 185
 - importowanie API, 192
- JAX-WS, 17, 20, 171
 - bezstanowe ziarna sesyjne, 180
 - dane uwierzytelnienia, 251
 - funkcjonalności, 173
- JaxWsProxyFactoryBean, 182, 183
- JBoss
 - dodatek do Maven, 61
 - instalacja narzędzi, 27
- JBoss AS, 13
 - definicja źródła danych, 113
 - instalacja sterownika JDBC, 112
 - deklaracja modułu, 112
 - instalacja modułu, 112
 - jako moduł, 112
 - integracja z Apache CXF, 173
 - JMS, 152
 - klastry aplikacji, 213
 - najczęściej używany serwer aplikacji, 18
 - podsystem komunikatów, 152
 - utworzenie źródła danych przy użyciu wiersza poleceń, 113
- JBoss AS 7, 17
 - adapter zasobów ActiveMQ, 168
 - alternatywne środowiska programistyczne, 28
 - Bean Pools, 158
 - bezpieczeństwo, 23
 - aplikacji, 239
 - Command Line Interface, 24
 - dowiązki własnych nazw JNDI do komponentów EJB, 64
 - efektywność, 21
 - implementacja CDI, 87
 - instalacja, 22
 - a systemy operacyjne, 23
 - narzędzia Maven, 29
 - serwera i komponentów klienta, 21
 - środowiska Eclipse, 26
 - IntelliJ IDEA, 29
 - interfejs zarządzania z poziomu wiersza poleceń, 21
 - jądro, 21
 - kategorie modułów, 87
 - klastry aplikacji, 214
 - Infinispan, 215
 - JGroups, 214
 - JSR - 107, 215
 - Multicast, 214
 - komunikacja SSL, 255
 - konfiguracja
 - domen, 216
 - wdrożenia bazującego na domenach, 32
 - konsola po uruchomieniu, 23
 - mechanizm wczytywania klas, 21
 - Modular Service Container, 21
 - moduły logowania, 242
 - nowe funkcje, 20, 31, 206
 - omówienie, 17
 - podstawowe koncepcje, 31
 - podsystem bezpieczeństwa, 241
 - połączenie z serwerem przy użyciu wiersza poleceń, 24
 - profile domen, 216
 - Profile/Container/EJB 3, 158
 - Profile/Messaging Provider, 153
 - przygotowanie środowiska, 21
 - instalacja Javy SE, 22
 - security-domain, 242
 - SSL, 255
 - system plików, 33
 - tryb
 - domenowy, 31
 - samodzielny, 31
 - uruchomienie, 23
 - architektura modułowa, 23
 - dostępność przez interfejs sieciowy, 24
 - dostosowanie właściwości początkowych, 24
 - konsoli webowej, 37
 - pamięć operacyjna, 24
 - ponowne, 26
 - usługi sieciowe
 - typu REST, 187
 - tworzenie, 175
 - usprawnienia zarządzania serwerem, 20
 - wdrażanie
 - aplikacji przy użyciu CLI, 202
 - do kilku węzłów, 203
 - jako foldery, 42
 - pierwszej aplikacji, 39
 - webowa konsola zarządzania, 21
 - wymagania sprzętowe, 21
 - wyświetlanie tajnego klucza, 218
 - zarządzanie serwerem, 36
 - zasady bezpieczeństwa, 260
 - zatrzymanie serwera, 25
 - na zdalnym systemie, 26
 - niezależny skrypt, 25
 - zdalne API dla CLI, 206
 - zestaw usług, 220
- JBoss Community, 28

- JBoss Forge, 117
 - aplikacja forge-demo, 272
 - Create New, 272
 - Search, 272
 - as7 deploy, 271
 - as7 undeploy, 272
 - budowanie i wdrożenie
 - aplikacji, 271
 - build, 271
 - cat, 269
 - dodatek AS 7 Forge, 271
 - forge.bat, 266
 - forge.sh, 266
 - forge-demo.var, 271
 - generate-entities, 269
 - help, 267
 - hibernate-tools, 268
 - instalacja, 265
 - konsola Forge, 266
 - list-commands, 266
 - new-project, 268
 - persistence.xml, 269
 - rozbudowa składni poleceń, 268
 - scaffold, 270
 - from-entity, 270
 - tworzenie aplikacji
 - pierwszej, 268
 - szybkie, 265
 - uruchamianie, 266
 - zalety, 268
- JBoss Microcontainer, 152
- JBoss Modules, 21, 34
- JBoss security, 239
- JBoss Tools, 27
 - instalacja manualna, 27
- JBoss WS-CXF, 173
- jboss.bind.address.management, 218
- jboss.domain.master.address, 218
- jboss.keystore, 256
- jboss.management.http.port, 36
- jboss.node.name, 215
- jboss.socket.binding.port, 215
- JBOSS_HOME, 34
- JBOSS_HOME/bin, 47
- jboss-as-controller-client, 209
- jboss-as-maven-plugin, 62, 91
- jboss-cli.bat, 24, 46, 114, 196
 - file, 203
 - gui, 197
- jboss-cli.sh, 24, 46, 114
 - file, 203
 - client.jar, 207
- jboss-dmr, 209
- jboss-ejb3.xml, 220, 250, 252
 - element dotyczący klastrów, 221
- jboss-ejb-api, 67
- jboss-ejb-client.properties, 70, 227, 261
 - uwierzytelnianie klienta, 74
- jboss-javaee-6.0-with-hibernate, 115
- jboss-javaee-6.0-with-tools, 115
- jboss-javaee6-webapp-ear-archetype, 114
- jboss-javaee6-webapp-archetype, 114
- jboss-jms-api_1.1_spec-1.0.0.Final.jar, 166
- jboss-remote-naming, 64
- jboss-remoting, 64
- jboss-web.xml, 244, 252
 - moduły logowania, 251
- JCA, 165
 - deskryptory, 166
 - współpraca z serwerem HornetQ, 152
- JDBC
 - instalacja w JBoss AS 7, 112
- JDO, 19
- JEE, 49
- JGroups, 214
- JMS, 17, 149, 150
 - cele, 150, 151
 - użycie, 155
 - dodanie producenta, 159
 - dostarczanie komunikatów, 151
 - tryb nietrwały, 151
 - tryb trwały, 151
 - elementy składowe, 151
 - kolejki, 150, 151
 - komunikat, 151
 - konsument komunikatów, 152
 - konsumpcja komunikatu, 150
 - sposób asynchroniczny, 151
 - sposób synchroniczny, 150
 - kontrola połączeń, 165
 - obiekt fabryki połączeń, 151
 - ogólna konfiguracja, 153
 - optymalizacja połączeń, 162
 - adapter zasobów, 163
 - podsystem komunikatów w JBoss, 152
 - połączenie, 152
 - producent komunikatów, 152
 - punkt-punkt, 150
 - selektory komunikatów, 163
 - sesja, 152
 - tematy, 150, 151
 - tworzenie i wykorzystanie fabryk, 153
 - użycie do integracji z innymi systemami, 165
 - wprowadzenie, 150
 - ziarna sterowane komunikatami, 156
 - obsługa żądania, 157
 - zserializowanie żądania, 157
- JMS Destinations, 155
 - Durable, 156
 - Selector, 156
- JMSService, 159
- JNDI, 64, 153
 - zmiana ustawień dowiązań, 154
- jndi-view, 199, 207
- JNP, 64
- JPA, 17, 110
 - API, 110
 - encje, 110
 - interfejs, 109
- JPA
 - JBoss Forge, 268
 - język zapytań, 110
 - korzystanie, 110
 - sterownik JDBC, 112
- JPA 2.0, 19
 - usprawnienia, 19
- JPA Entities from Table, 117
- JPA Query Language, 19
- JPQL, 19, 110
- jQuery, 102
- JSF, 17, 20, 83, 95
 - cykl życia, 95
 - tworzenie widoków, 128
 - warstwa, 119
- JSF 2.0, 99, 128
- JSF 2.0, 18
 - usprawnienia, 19

- JSON, 189
 - gramatyka, 189
 - JSON.simple, 191
 - JSONArray, 191
 - JSP, 17, 95
 - cykl życia, 95
 - getUserPrincipal(), 241
 - isUserInRole(), 240
 - tworzenie stron, 96
 - JSR 224, 171
 - JSR 311, 171
 - JSR-107, 215
 - JSR-250, 55, 90
 - JSR-299, 20
 - JSR-303, 19
 - JSR-303 Bean Validation, 115, 118
 - JSR-311, 185
 - JSSE, 255
 - JTA, 139
 - junit, 139
 - JVM, 32
 - python, 206
 - python, 206
 - definiowanie struktury kodu, 207
 - instalacja, 206
 - python script.py, 207
- K**
- KeepAliveTimeout, 234
 - keep-content, 48
 - Kepler, 26
 - keystore, 255, 260
 - keytool, 255
 - klasa
 - pojedyncze instancje klas, 120
 - produkująca
 - dodawanie, 120
 - kod zapytań, 123
 - klasy
 - affinity, 222
 - aplikacji webowych
 - równoważenie obciążenia, 232
 - tworzenie, 231
 - domenowy, 215
 - domain.xml, 216
 - host.xml, 216
 - uruchamianie, 216
 - dziennik zdarzeń, 220, 229
 - EJB, 221
 - system rezerwacji biletów, 223
 - wdrożenie i testowanie
 - wysokiej dostępności, 228
 - zamiana pamięci
 - podręcznej na wersję rozproszoną, 225
 - zdalny klient świadomy
 - istnienia klastra, 227
 - ziarna sesyjne o wysokiej dostępności, 222
 - instancja serwera, 213
 - JVM, 224
 - konfiguracje hostów, 218
 - moduły, 216
 - obsługa żądania, 222
 - pamięć podręczna, 224
 - Infinispan, 224
 - JPA, 224
 - SingletonService, 224
 - podstawy, 213
 - równoważenie obciążenia, 214
 - samodzielny, 215
 - standalone.bat, 215
 - standalone.sh, 215
 - uruchamianie, 215
 - skalowalność, 214
 - stanowe ziarna sesyjne, 222
 - w JBoss AS 7, 214
 - architektura, 214
 - wdrażanie aplikacji, 220
 - węzły
 - tworzenie połączeń, 222
 - wysoka dostępność, 214
 - zachowanie spójności
 - aplikacji, 215
 - zamiana pamięci podręcznej na wersję rozproszoną, 225
 - Entity, 269
 - JaxWsProxyFactoryBean, 182, 183
 - MDBService, 158
 - MessageDigest, 248
 - ModelNode, 210
 - POJOWebService, 175
 - PollerBean, 105
 - RemoteEJBClient, 68
 - Resources, 120
 - scriptsupport.CLI, 206
 - Seat, 95, 118, 180
 - SeatProducer, 120, 121
 - SeatType, 117, 118
 - SeatTypeProducer, 120, 121
 - TheatreBookerBean, 60
 - TheatreBox, 55, 180
 - TicketController, 127
 - TicketRESTService, 188
 - TicketService, 123, 142
 - TicketSOAPSService, 180, 181
 - TicketTest, 141, 146
 - TicketWebServiceIT, 182
 - TimerConfig, 75
 - URLConnection, 190
 - klient
 - wiersza poleceń, 36
 - zarządzania, 209
 - zdalny świadomy istnienia klastra, 227
 - klucz
 - prywatny, 253
 - publiczny, 253
 - tajny, 253
 - kod zapytań
 - tworzenie, 123
 - kolejki, 150, 151
 - TicketQueue, 166
 - komponenty
 - Enterprise JavaBeans, 240
 - końcówka nasłuchująca
 - serwera, 174
 - webowe, 240
 - model bezpieczeństwa, 239
 - komponenty EJB, 49, 107
 - a ziarna zarządzane
 - przez JSF, 107
 - a ziarno CDI, 125
 - asynchroniczne, 50
 - bezinterfejsowy, 50
 - bezstanowe ziarna sesyjne, 49

- komponenty EJB
 - BookerService, 123
 - dodanie metod
 - asynchronicznych, 77
 - dostępność dla zdalnych
 - klientów, 249
 - dowiązanie JNDI, 65
 - klastrowe, 221
 - nowe funkcjonalności, 49
 - rodzaje, 49
 - singletonowe, 50
 - dodanie ziarna sesyjnego, 60
 - dodanie ziarna
 - bezstanowego, 59
 - konfiguracja pliku pom.xml, 54, 67
 - Maven, 51
 - przygotowanie ziaren
 - sesyjnych, 58
 - sterowanie współbieżnością
 - ziarna, 57
 - tworzenie, 50
 - tworzenie kodu aplikacji, 55
 - stanowe ziarna sesyjne, 50
 - TicketService, 123
 - transakcyjne, 125
 - trwałość, 120
 - udostępnianie lokalnym
 - klientom, 59
 - usługa czasomierza, 75
 - użycie wyjątku wykonania, 61
 - w niezaufanych sieciach, 255
 - wdrożenie aplikacji EJB, 61
 - wstrzykiwanie
 - do klasy, 59
 - do ziarna CDI, 93
 - fabryki połączeń, 154
 - wywołanie z poziomu klienta
 - zdalnego, 59
 - zabezpieczenia, 249
 - zdalny klient, 64
 - dowiązania JNDI, 64
 - ziarna sterowane
 - komunikatami, 50
 - komponenty JavaBeans, 118
 - komunikacja
 - dostawca i konsument usług, 172
 - EJB, 258
 - zabezpieczanie, 258
 - komponenty EJB
 - filtrowanie komunikatów, 163
 - HTTP, 255, 256, 257
 - komponenty EJB
 - JMS, 165
 - publikuj-subskrybuj, 151
 - konsument komunikatów, 152
 - producent komunikatów, 152
 - punkt-punkt, 151
 - konsument komunikatów, 152
 - producent komunikatów, 152
 - selektory, 163
 - SSL, 255, 260
 - komunikat JMS
 - elementy, 151
 - konektor
 - HTTP, 198
 - odczytanie wartości
 - atrybutu, 199
 - ustawianie portu, 199
 - remoting, 261
 - konfiguracja
 - bazy danych, 111
 - hostów, 218
 - kontroler domeny, 216
 - pom.xml, 54
 - projektu klienta, 67
 - testowanie aplikacji, 139
 - serwera, 198
 - trwałości, 119
 - wdrożenia bazującego
 - na domenach, 32
 - konsola webowa
 - uruchomienie, 37
 - zakładka
 - Profile, 38, 45
 - Runtime, 38, 43
 - zarządzanie wdrożeniami, 43
 - przyciski, 44
 - skaner wdrożeń, 45
 - konsument komunikatów, 152
 - kod, 156
 - konsumpcja komunikatu, 150
 - ActiveMQ, 168
 - kontekstowe działanie, 84
 - kontenery
 - komponentów, 240
 - typy bezpieczeństwa, 240
 - MDB, 157
 - weld, 140
 - kontroler
 - domeny, 31, 217
 - konfiguracja, 216
 - uruchamianie, 218
 - wskazanie nazwy
 - użytkownika, 219
 - hosta, 32, 218
 - procesów, 32
 - sterujące żądaniami
 - użytkowników
 - dodanie, 126
 - konwencje, 15
 - konwersacja
 - miejsce rozpoczęcia, 128
 - zakres, 126
 - końcówka nasłuchująca serwera, 174
 - kryptografia
 - asymetryczna, 253
 - klucza publicznego, 253
- L**
- lib/ext, 35
 - lista, 73
 - list-commands, 266
 - load balancing, 214
 - log, 35
 - login.jsf, 245
 - login-config, 245
 - login-module, 242
 - lookup, 226
 - ls, 199
- Ł**
- łączenie
 - ze zdalnymi hostami, 196
- M**
- M2_HOME, 29
 - mad-in-the-middle, 256
 - main, 137

- main-server-group, 219
- Manage Deployments, 43
- Management User, 36
- ManagementRealm, 36
- Manager, 242, 247
- ManagerBalancerName, 234
- Maven, 29
 - Arquillian, 137
 - Create a simple project, 52
 - dodanie konfiguracji, 115
 - dodatki
 - exec, 229
 - JBoss, 61
 - uruchomienie zdalnego klienta EJB, 71
 - instalacja, 29
 - test, 30
 - integracja z Eclipse, 51
 - konfiguracja pliku pom.xml, 54
 - kreator, 51
 - pisanie testu Arquillian, 137
 - Project View, 54
 - projekt, 114
 - tworzenie, 114
 - zaimportowanie, 138
 - Run As/JUnit Test, 144
 - Source folder, 71
 - użycie zarządzanego kontenera, 144
 - wdrożenie aplikacji, 63, 132
 - klastrowej, 228
 - widok Project Explorer, 139
 - zalety stosowania, 51
 - Maven Integration for Eclipse, 51
 - Maven Project, 51, 65, 88
 - maven-compiler-plugin, 62
 - maven-ejb-plugin, 62
 - maven-failsafe-plugin, 184
 - maven-surefire-plugin, 184
 - Max Pool Size, 158
 - MaxKeepAliveRequests, 234
 - max-pool-size, 163
 - MDB, 50, 156
 - MDBService, 158
 - mechanizm
 - BASIC, 243
 - bezpieczeństwa, 36, 239
 - ApplicationRealm, 74
 - Extension-List, 35
 - HTTP Digest, 36
 - JAXB, 174
 - JAX-WS, 174
 - selektora, 222
 - uwierzytelniania, 36
 - MessageDigest, 248
 - Message-Driven Beans, 50
 - messages, 98
 - Messaging Provider, 155
 - META-INF, 88, 167, 240
 - META-INF/MANIFEST.MF, 209, 236
 - metoda
 - adnotacje bezpieczeństwa, 250
 - EJB
 - uwierzytelnianie, 249
 - logowania w konsoli serwera aplikacji, 147
 - obserwująca, 94
 - obsługi obserwatora, 122
 - produkująca, 94, 121
 - przeprowadzająca test, 141
 - zwrotna
 - harmonogram wywołań, 76
 - metody
 - addPackage, 141
 - asInt, 209
 - automaticCustomer, 77
 - bookSeat, 99, 125, 182, 250
 - bookSeatAsync, 78, 79
 - buyTicket, 57, 58, 77, 189, 191
 - calculatePower, 175
 - cancelTimers, 77
 - create, 141, 184
 - createSeatType, 125, 142
 - createTheatre, 125
 - DELETE, 186
 - deleteAllData, 123
 - doCleanUp, 125
 - findSeat, 77
 - finish, 130
 - get, 226
 - GET, 186
 - getSeatList, 57, 189
 - getSeatPrice, 57
 - getSeats, 94, 182
 - getType(), 209
 - getUser, 187
 - handleGETRequest, 186
 - handlePOSTRequest, 187
 - isDone, 80
 - onMessage, 158
 - onMessage(), 151, 157
 - POST, 186
 - pressAKey, 227
 - put, 226
 - PUT, 186
 - receive(), 150
 - removeAttribute, 235
 - restart, 128
 - retrieveData, 122
 - Seat.class.getPackage(), 141
 - sendMessage, 163
 - setAddress, 183
 - setAttribute, 235
 - setProxyClass, 183
 - setupTheatre, 56, 75
 - start, 226
 - testTicketAgency, 142, 146
 - timeout, 75
 - mgmt-users.properties, 15, 36
 - middleware, 11
 - min-pool-size, 163
 - mod_advertise, 234
 - mod_cluster, 231, 232
 - archiwum, 233
 - domyślna konfiguracja, 237
 - instalacja, 233
 - zalety, 232
 - mod_jk, 232
 - mod_manager, 234
 - mod_proxy, 232, 234
 - model
 - punkt-punkt, 150
 - ModelNode, 209, 210
 - get, 209
 - Modular Service Container, 21
 - module, 204
 - module.xml, 112
 - module-name, 64
 - modules, 34
 - modul
 - com.mysql, 112
 - dodawany warunkowo, 87
 - hibernate-tools, 268
 - klastra, 216
 - jawnie włączony w opisie wdrożenia aplikacji, 88
 - Login, 240
 - login-module, 242
 - logowania, 263
 - Database, 242, 246, 249
 - konfiguracja, 242

- modul
 - RealmUserRoles, 243
 - RealmUsersRoles, 241, 242
 - role, 244
 - sekwencja konfiguracji, 244
 - szyfrowanie hasel, 247
 - UserRoles, 246
 - w aplikacji systemu rezerwacji biletów, 243
 - wykorzystujący bazę danych, 246
 - mod_advertise, 234
 - mod_cluster, 231, 232
 - instalacja, 233
 - zalety, 232
 - mod_jk, 232
 - mod_manager, 234
 - mod_proxy, 232, 234
 - mod_proxy_ajp, 234
 - niejawnie dodawany
 - do aplikacji, 87
 - rozdzielający, 174
 - uwierzytelnienia
 - w domenie bezpieczeństwa, 242
 - modułowy mechanizm
 - wczytywania klas, 21, 34
 - money, 98
 - MSC, 21
 - multicast, 214
 - mod_cluster, 232
 - mvn clean install, 143
 - mvn exec:exec, 230
 - mvn install, 61, 101, 132
 - mvn install exec:exec, 72
 - mvn install jboss-as:deploy, 177, 189
 - mvn jboss-as:deploy, 62, 101, 133
 - mvn verify, 185, 192
 - mvn --version, 30
 - MySQL, 111
 - sterownik JDBC, 112
 - mysqldomain, 252
- ## N
- nagłówki komunikatu, 151, 163
 - name, 120, 176
 - Name, 45
 - naming, 199
 - narzędzia
 - add-user.bat, 36
 - add-user.sh, 36
 - Arquillian, 136
 - Browse workspace, 62
 - CLI, 197
 - do testowania usług sieciowych, 178
 - do zarządzania certyfikatami, 255
 - informacji o adapterze zasobów, 166
 - IronJacamar, 166
 - keytool, 255
 - Maven, 29, 51
 - rar-info.sh, 166
 - soapUI, 178
 - twiddle, 24
 - zarządzania serwerem, 35
 - nawigacja niejawna, 128
 - nazwa-operacji, 198
 - nazwa-węzła, 198
 - NetBeans, 29
 - Netty, 152, 153
 - New Input-Output, 152
 - New soapUI Project, 179
 - new-project, 268
 - Nie istnieje, 157
 - NIO, 152
 - no-interface, 59
 - non-heap-memory-usage, 208
 - NULLIF, 19
- ## O
- obiekty
 - dataTable, 98
 - Executor, 79
 - fabryka połączeń, 151, 153
 - Future, 78
 - futureResult, 80
 - javax.enterprise.event.Event, 94
 - keystore, 260
 - konsument komunikatów, 152
 - ModelNode, 209, 210
 - naśladowące, 136
 - POJO, 152
 - pośredniczące dla interfejsu JAX-WS, 182
 - producent komunikatów, 152
 - Runnable, 79
 - Seat, 56, 57
 - SeatType, 127
 - Theatre, 146
 - TheatreInfoBean, 98
 - Timer, 77
 - truststore, 259
 - obserwatory, 94
 - warunkowe, 94
 - obsługa
 - wiadomości SOAP, 174
 - współbieżności, 58
 - zadań, 50
 - klastery, 222
 - odzworowanie akcji CRUD, 186
 - ograniczenia
 - bezpieczeństwa, 244, 250
 - tworzenie, 118
 - onListChanged, 122
 - onMessage, 158
 - onMessage(), 151, 157
 - operacje na zasobach, 198
 - attribut węzła, 199
 - lista
 - dostępnych zasobów i węzłów, 198
 - dowiązań JNDI, 199
 - operacje specjalne, 199
 - ustawianie portu konektora HTTP, 199
 - optymalizacja połączeń JMS, 162
 - org.hibernate, 87
 - org.jboss
 - .arquillian.junit, 139
 - .arquillian.protocol, 140
 - .as.cli.scriptsupport.CLI, 207
 - .as.web, 87
 - .as.weld, 87
 - .ejb3.annotation.
 - SecurityDomain, 250
 - .marshalling, 67
 - .msc.service.Service, 224
 - .remoting3, 67
 - .resteasy, 87
 - .sas, 67
 - .xnio, 67
 - org.mysql, 205
 - organ certyfikujący, 254
 - ORM, 110
 - other-server-group, 220

P

- pamięć podręczna, 224
 - infinispan, 225, 229
 - javax.cache.Cache, 225
 - pobieranie instancji, 225
 - rozproszona
 - operacje, 226
 - zamiana na pamięć rozproszoną, 225
- para kluczy, 253
 - generowanie
 - dla klienta, 259
 - dla serwera, 259
- parametry
 - b, 215
 - c, 215
 - host-config, 218
 - jboss.node.name, 215
 - jboss.socket.binding.port, 215
- Password, 37
- Path, 45, 189
- PATH, 30
- pełna publikacja, 42
- pełne wsparcie biznesowe, 22
- PermitAll, 249
- Persistence API, 115
- persistence.xml, 119, 269
 - dostawca trwałości, 120
 - źródło danych, 120
- perspektywa soapUI, 179
- PicketBox, 239
- picketbox-4.0.7.Final.jar, 248
- Plain Old Java Object, 109
- Plain Old Java Objects, 152
- pliki
 - activemq-rar-5.7.0.txt, 166
 - add-user.bat, 217
 - add-user.sh, 217
 - application-roles.properties, 242
 - application-users.properties, 242
 - arquillian.xml, 142, 145
 - beans.xml, 88
 - certreq.csr, 257
 - definicji schematów XML, 34
 - domain.xml, 113, 216, 217, 233
 - ejb-jar.xml, 54, 250, 252
 - ejb-roles.properties, 260
 - ejb-users.properties, 260
 - HelloWorld.war.dodeploy, 42
 - HelloWorld.war.failed, 42
 - host.xml, 216, 217, 218, 219
 - httpd.conf, 234
 - ironjacamar.xml, 166
 - JAR klienta, 34
 - jboss - ejb3.xml, 220
 - jboss.keystore, 256
 - jbossas-web_1_1.xsd, 41
 - jboss-cli.bat, 46
 - jboss-cli.sh, 46
 - jboss-cli-client.jar, 207
 - jboss-ejb3.xml, 221, 250, 252
 - jboss-ejb-client.properties, 70, 227, 261
 - jboss-jms-api_1.1_spec-1.0.0.Final.jar, 166
 - jboss-web.xml, 244, 251, 252
 - keystore, 255
 - konfiguracyjne
 - trybu domenowego, 31
 - trybu samodzielnego, 31
 - META-INF/MANIFEST.MF, 88, 209, 236
 - mgmt-users.properties, 36
 - module.xml, 112
 - persistence.xml, 119, 269
 - picketbox-4.0.7.Final.jar, 248
 - pom.xml, 54
 - konfiguracja, 54, 67
 - ra.xml, 167
 - repozytorium kluczy, 256
 - script.py, 207
 - setup.xhtml, 128
 - signed_ca.txt, 258
 - standalone - full -ha.xml, 233
 - standalone.conf, 24
 - standalone.conf.bat, 24
 - standalone.xml, 113
 - standalone-full.xml, 154
 - standalone-ha.xml, 233
 - uruchomieniowy IDE
 - Eclipse, 26
 - WAR, 201
 - web.xml, 187, 220, 232, 251
- podfoldery
 - data, 35
 - log, 35
 - tmp, 35
- podpisanie certyfikatu, 257
- podstawowe koncepcje, 31
- podsystem
 - bezpieczeństwa, 241
 - komunikatów, 152
 - korzystanie, 155
- POJO, 109, 110, 134, 152
- POJOService, 177
- POJOWebService, 175, 251
- pole
 - description, 119
- polecenia
 - ./jboss-cli.sh, 196
 - as7 deploy, 271
 - as7 undeploy, 272
 - begin(), 125
 - build, 271
 - cat, 269
 - cd, 199
 - CLI, 197
 - commit(), 125
 - connect, 25, 196, 207
 - deploy, 46, 47, 199, 201
 - e-mail, 79
 - execute, 210
 - forge.bat, 266
 - generate-entities, 269
 - get, 209
 - help, 267
 - Install New Software, 27
 - java -version, 22
 - jboss-cli.bat, 196
 - jboss-cli.bat --file, 203
 - jboss-cli.bat --gui, 197
 - jboss-cli.sh --file, 203
 - jndi-view, 199, 207
 - JPA Entities from Table, 117
 - python, 206
 - python script.py, 207
 - lista, 73
 - list-commands, 266
 - ls, 199
 - module, 204
 - mvn clean install, 143
 - mvn exec:exec, 230
 - mvn install, 61, 101, 132
 - exec:exec, 72
 - jboss-as:deploy, 177, 189
 - mvn jboss-as:deploy, 62, 101, 133
 - mvn verify, 185, 192

- polecenia
 - mvn --version, 30
 - New/Dynamic Web Project, 39
 - New/Servlet, 40
 - new-project, 268
 - rar-info, 166
 - read-attribute, 199
 - read-resources, 198
 - reload, 26
 - rezerwuj, 73
 - rezerwujasync, 79
 - rollback(), 125
 - Run Configurations, 62
 - run.bat, 23
 - run.sh, 23
 - scaffold, 270
 - scaffold from-entity, 270
 - shutdown, 25, 196
 - standalone.bat, 23, 215
 - standalone.sh, 215
 - standalone.sh –c standalone-full.xml, 155, 161
 - undeploy, 47, 48, 199, 201
 - write-attribute, 199
 - PollerBean, 104
 - połączenie, 152
 - pamięć podręczna, 162
 - z serwerem przy użyciu wiersza poleceń, 24
 - pom.xml
 - API EJB 3.1, 55
 - API JBoss JMS, 161
 - API JBoss logging, 55
 - biblioteka JSON.simple, 192
 - Common Annotations API, 55
 - dodanie
 - JBoss do Maven, 61
 - zależności BOM, 67
 - kompilacja kodu klienta, 184
 - konfiguracja, 54, 67, 139
 - referencja do Bill of Material, 54
 - wskazanie listy bibliotek, 90
 - zależności, 116
 - port-offset, 220
 - POST, 186
 - pressAKey, 227
 - price, 169
 - proces JVM, 32
 - procesor adnotacji, 116
 - producent komunikatów, 152
 - fabryka połączeń stosująca pulę obiektów, 162
 - producenty, 121
 - encji, 121
 - producer, 93
 - profil, 140
 - arq-jbossas-managed, 140
 - arq-jbossas-remote, 140
 - Profile, 38, 45
 - profile domen, 216
 - default, 216
 - full, 216
 - full - ha, 216
 - ha, 216
 - Project Facets, 99
 - projekt
 - Forge, 268
 - jboss-remote-naming, 64
 - jboss-remoting, 64
 - JNP, 64
 - Maven, 51, 65, 88, 114
 - rozdział06, 138
 - Seam, 87
 - ShrinkWrap, 141
 - ticket-agency-test, 138
 - ticket-agency-ws, 175
 - webapp-javaee6, 161
 - projektowanie widoków, 99
 - protokół
 - HTTP, 149, 249
 - Multicast, 214
 - przesyłanie komunikatów, 149
 - RMI, 149
 - RMI-IIOP, 249, 258
 - SSL, 252
 - TLS, 252
 - provided, 55, 192
 - przełączanie na inną konfigurację serwera, 161
 - przestrzeń nazw JNDI, 113
 - przesyłanie komunikatów, 149
 - konsumpcja komunikatów, 150
 - proces, 150
 - subskrypcja
 - nietrwała, 150
 - trwała, 150
 - przykłady dla książki, 16
 - PTP, 151
 - pub-sub, 151
 - Pula gotowych metod, 157
 - pula połączeń
 - wielkość, 163
 - pule instancji MDB
 - konfiguracja rozmiaru, 158
 - wielkość maksymalna, 158
 - punkt-punkt, 150, 151
 - put, 226
 - PUT, 186
- ## Q
- QueueReceiver, 152
 - QueueSender, 152
- ## R
- ra.xml, 167
 - rar-info.sh, 166
 - znajdowanie problemów, 166
 - READ, 58
 - read-attribute, 199
 - read-resources, 198
 - Realm, 36
 - RealmUserRoles, 243
 - RealmUsersRoles, 241, 242
 - receive(), 150
 - Red Hat Enterprise Platform, 22
 - reguły uwierzytelniania, 260
 - reload, 26
 - w głównej ścieżce węzła, 26
 - Remote Method Invocation, 149
 - Remote Procedure Call, 171, 176
 - remote.connection.nodeX.host, 228
 - remote.connectionprovider.
 - create.options.org.xnio.Options.SSL_ENABLED, 71
 - remote.connections, 71
 - RemoteConnectionFactory, 153
 - RemoteEJBClient, 68
 - remoting, 261
 - removeAttribute, 235
 - replikacja stanów sesji HTTP
 - w pamięci, 235
 - repozytorium kluczy, 255, 256

- reprezentacja JSON, 189
 - Request 1, 180
 - RequestScope, 126
 - RequestScoped, 84, 86
 - resources, 137
 - archetyp Maven, 71
 - Resources, 120
 - zasoby, 121
 - rest, 190
 - REST, 20, 185
 - asercja JUnit, 192
 - dostęp do zasobów, 186
 - framework kliencki JAX-RS, 187
 - odzworowanie typu jedno-
do-jednego, 186
 - pobranie listy miejsc, 190
 - testowanie, 191
 - tworzenie, 187
 - pojedynczych żądań HTTP, 190
 - usługa JAX-RS, 190
 - użycie usługi, 190
 - wiele usług tej samej aplikacji, 190
 - zastosowanie, 193
 - żądania
 - ClientRequest, 191
 - typu GET, 189
 - restart węzłów aplikacji serwerowej, 204
 - RESTEasy, 187
 - aktywacja, 187
 - API klienta, 192
 - retrieveData, 122
 - rezerwuj, 73
 - rezerwujasync, 79
 - RichFaces, 102
 - instalacja, 103
 - RMI, 149, 255
 - RMI-IIOP, 261
 - RMI-IIOP, 249, 258
 - role
 - bezpieczeństwa, 250
 - ejbRole, 262
 - użytkowników, 242, 247
 - rollback(), 125
 - rozbudowa testu, 145
 - rozdział06, 138
 - równoważenie obciążenia, 214, 232
 - aplikacje zgodne z klastrami, 235
 - instalacja mod_cluster, 233
 - wysoka dostępność
 - w aplikacjach JSF, 235
 - zagadnienia
 - programistyczne, 235
 - RPC, 171, 176
 - RSA, 256
 - Run Configurations, 62
 - run.bat, 23
 - run.sh, 23
 - Runnable, 79
 - Runtime, 38, 43
- ## S
- SASL, 262
 - SASL_POLICY_NOANONYMOUS, 262
 - scaffold, 270
 - scaffold from-entity, 270
 - Scan Interval, 46
 - schema, 256
 - script.py, 207
 - scriptsupport.CLI, 206
 - Seam, 87
 - seat, 190
 - Seat, 56, 57, 95, 118, 180
 - SEAT, 111
 - Seat.class.getPackage(), 141
 - seat.Id, 97
 - seat_id, 111, 118
 - SEAT_TYPE, 111
 - odzworowanie, 117
 - seatId, 99, 169
 - SeatProducer, 120, 121, 122
 - SeatType, 117, 118, 119, 127, 142
 - SeatTypeProducer, 120, 121
 - seatTypes, 122
 - secure, 256
 - security-constraints, 243, 251
 - security-domain, 242, 263
 - SecurityDomain, 250
 - SEI, 180
 - Selector, 156
 - selektory komunikatów, 163
 - sendMessage, 163
 - HIGH, 164
 - LOW, 164
 - String, 163
 - serializacja, 235
 - ServerAdvertise, 235
 - server-groups, 48, 202
 - servers, 35
 - Service, 176
 - name, 176
 - Service Endpoint Interface, 180
 - serviceName, 176
 - serwer
 - Apache
 - konfiguracja, 233
 - aplikacji
 - instalacja, 21
 - jako część zarządzanej domeny, 20
 - JBoss AS, 18
 - zarządzanie, 195
 - komunikatów, 149
 - równoważenia obciążenia HTTP, 231
 - sesja, 152
 - pamięć podręczna, 162
 - sesje przyklepione, 237
 - SessionScoped, 86
 - setAddress, 183
 - setAttribute, 235
 - setProxyClass, 183
 - setup.xhtml, 128
 - setupTheatre, 56, 75
 - SFSB, 50
 - SFSB TheatreBroker, 70
 - ShrinkWrap, 141
 - shutdown, 25, 196
 - signed_ca.txt, 258
 - Simple Authentication and Security Layer, 262
 - singletonowy EJB, 50, 51, 223, 224
 - SingletonService, 224
 - singletony, 50
 - skaner wdrożeń
 - atrybuty, 45
 - zmiana właściwości, 45
 - skrótów klawiaturowe
 - Ctrl+C, 25
 - Ctrl+Spacja, 99
 - skrypty CLI, 203
 - dodawanie zasobów JMS, 205
 - instalacja źródła danych jako modułu, 204
 - Jython, 206
 - analiza zmiennej odpowiedzi, 208

- skrypty CLI
 - pobranie zawartości zasobów, 208
 - zwrócenie widoku JNDI serwera aplikacji, 207
 - restart serwerów w domenie, 204
 - użycie zaawansowanych języków, 205
 - SLSB, 49, 180
 - SLSB TheatreInfo, 70
 - słowa kluczowe
 - synchronized, 58
 - SOAP, 172
 - moduł rozdzielający, 174
 - odpowiedź, 174
 - HTTP, 174
 - wiadomość, 172
 - obsługa, 174
 - styl dokumentowy, 177
 - w stylu RPC, 177
 - zamiana na obiekt Javy, 174
 - zastosowanie, 193
 - soapUI, 178
 - socket-binding, 256
 - Spring, 152
 - Spring 3, 84
 - ssl, 256
 - SSL, 252
 - format x.509, 254
 - komunikacja, 255
 - dla połączeń z EJB, 260
 - narzędzia do zarządzania certyfikatami, 255
 - nawiązywanie połączenia, 259
 - repozytorium kluczy, 255
 - uwierzytelnianie wzajemne, 254
 - zasada bezpieczeństwa, 260
 - po stronie klienta, 261
 - SSL_ENABLED, 262
 - sslPublicKey.cer, 259
 - stan EJB
 - @Clustered, 222
 - @Stateful, 222
 - standalone, 34
 - struktura zawartości, 34
 - standalone.bat, 23, 215
 - standalone.conf, 24
 - standalone.conf.bat, 24
 - standalone.sh, 215
 - standalone.sh -c standalone-full.xml, 155, 161
 - standalone.xml, 113
 - standalone-full.xml, 154
 - standalone-full-ha.xml, 215
 - mod_cluster, 233
 - standalone-ha.xml, 215
 - mod_cluster, 233
 - standardy
 - dokumentów WSDL, 172
 - usług sieciowych, 171
 - stanowe ziarna sesyjne, 50
 - stanowy EJB, 223
 - start, 226
 - STARTED, 204
 - STARTTTL, 262
 - Stateful Session Beans, 50
 - Stateless Session Beans, 49
 - sterowanie współbieżnością ziarna, 57
 - sterownik JDBC, 112, 205
 - wskazanie ścieżki, 112
 - strona
 - JSF
 - tworzenie, 96
 - wywoływanie, 100
 - strona
 - login.jsf, 245
 - logowania, 246
 - szablonu, 131
 - subskrypcja
 - nietrwała, 150
 - trwała, 150
 - synchronized, 58
 - system
 - plików, 33
 - połączeń zdalnych
 - domyślny port, 71
 - rezerwacji biletów
 - dodanie zależności, 90
 - przekształcenie, 88
 - zarządzania, 210
 - szablony, 96
 - szybkie środowisko, 213
 - szyfrowanie, 252
 - asymetryczne, 253
 - problemy, 253
 - haseł, 247
 - symetryczne, 253
 - z kluczem publicznym, 253
- ## Ś
- środowisko programistyczne alternatywne, 28
- ## T
- tabela
 - SEAT, 111
 - SEAT_TYPE, 111, 117
 - user, 268
 - tajny klucz, 253
 - Target Runtime, 39
 - targetNamespace, 176
 - tematy, 150, 151
 - testowanie
 - aplikacji, 135
 - funkcji typu enterprise, 139
 - testTicketAgency, 142, 146
 - testy, 135
 - integracyjne, 135
 - dla aplikacji Javy, 136
 - maven-failsafe-plugin, 184
 - polecenia, 185
 - jednostkowe, 135
 - maven-surefire-plugin, 184
 - obiekty naśladujące, 136
 - logi, 144
 - narzędzia pomagające, 136
 - prostej usługi sieciowej, 178
 - rozbudowa, 145
 - TicketTest, 143
 - uruchomienie
 - na zdalnym serwerze, 144
 - w zarządzanym kontenerze, 144
 - Theatre, 146
 - TheatreBookerBean, 60, 91, 250
 - dowiązanie JNDI, 65
 - TheatreBox, 55, 91, 180, 236
 - Infinispan, 225
 - TheatreInfo
 - dowiązanie JNDI, 65
 - TheatreInfoBean, 93, 98
 - ticket-agency-cdi, 89
 - ticket-agency-ejb, 53, 65, 67
 - ticket-agency-jpa, 143
 - ticket-agency-test, 138
 - ponowna kompiacja, 143
 - uruchomienie TicketTest, 143
 - ticket-agency-test.war, 141

- ticket-agency-ws, 175, 178
 - żądanie HTTP typu GET, 190
 - TicketController, 126, 127, 130
 - ticketQueue, 159
 - TicketQueue, 166
 - TicketRETSERVICE, 188, 190
 - TicketService, 123, 142
 - TicketSOAPSERVICE, 180, 181
 - TicketSOAPSERVICEItf, 180
 - ticketssystem, 111
 - TicketTest, 141, 146
 - uruchomienie, 143
 - przy użyciu profilu
 - zdalnego Arquillian, 143
 - TicketWebServiceIT, 182
 - timeout, 75
 - Timer, 77
 - TimerConfig, 75
 - TimerService, 75
 - TLS, 252
 - tmp, 35
 - TopicPublisher, 152
 - TopicSubscriber, 152
 - tożsamość serwera, 260
 - transmisja do wielu celów, 214
 - treść komunikatu, 151
 - truststore, 259
 - zaimportowanie certyfikatu, 260
 - trwałość danych
 - dodanie do aplikacji, 111
 - dostawca trwałości, 120
 - konfiguracja, 119
 - minimalne wymagania dla klasy, 110
 - standardy, 109
 - tryby
 - domenowy, 31
 - wdrażanie aplikacji, 35
 - wdrażanie aplikacji za pomocą narzędzia CLI, 47
 - działania JBoss AS 7, 31
 - system plików, 33
 - zarządzanie, 36
 - nietrwały, 151
 - samodzielny, 31
 - konsola webowa, 38
 - trwałe, 151
 - wsadowy, 21
 - TTLS, 262
 - Tunneled Transport Layer Security, 262
 - twiddle, 24
 - tworzenie
 - aplikacji
 - wykorzystujących JBoss JMS Provider, 149
 - czasomierza programowego, 75
 - encji, 116
 - fabryk połączeń, 153
 - klas EJB, 55
 - klastrowej wersji systemu rezerwacji biletów, 223
 - klienta zdalnego świadomego istnienia klastra, 227
 - kodu
 - aplikacji EJB, 55
 - klienta EJB, 68
 - zapytań, 123
 - ziaren, 91
 - ograniczeń w formie adnotacji, 118
 - repozytorium kluczy, 255
 - singletonowych komponentów EJB, 50
 - stanowych ziaren sesyjnych o wysokiej dostępności, 222
 - szablonów, 96
 - widoków w JSF, 95
 - zdalnego klienta EJB, 64
 - ziaren sterowanych komunikatami, 158
 - typ użytkownika, 36
 - typ-węzła, 198
- ## U
- ui:composition, 98
 - Ultimate edition, 28
 - undeploy, 47, 48, 199, 201
 - undeploy --all-relevant-server-groups, 202
 - Unified Expression Language, 85
 - UPDATE, 186
 - URL *.jsf, 100
 - URLConnection, 190
 - uruchamianie
 - Forge, 266
 - konsoli webowej, 37
 - Use a custom deploy folder, 43
 - Use default location, 39
 - Username, 37
 - UserRoles, 246
 - usługa
 - CRON, 76
 - czasomierza EJB, 75
 - dodanie do aplikacji, 123
 - usługi sieciowe, 171
 - a integracje JMS i JCA, 165
 - adnotacje dla klasy, 176
 - adres końcówki, 178
 - bazujące na EJB, 251
 - bazujące na REST, 185
 - aktywacja RESTEasy, 187
 - dodanie obsługi REST do aplikacji, 187
 - dostęp do zasobów, 186
 - kompilacja przykładu, 192
 - użycie, 190
 - w JBoss, 187
 - bazujące na SOAP, 172
 - architektura JAX-WS, 174
 - bezstanowe ziarna sesyjne EJB 3, 180
 - kompilacja przykładu, 184
 - końcówka nasłuchująca serwera, 174
 - podejście z dołu do góry, 173
 - podejście z góry na dół, 173
 - sprawdzenie z poziomu konsoli, 178
 - stos usług w JBoss, 173
 - strategię tworzenia, 172
 - testowanie, 178
 - tworzenie klienta, 182
 - tworzenie w JBoss AS 7, 175
 - wykonanie usługi dla POJO, 175
 - bezpieczeństwo, 251
 - działanie, 171
 - interceptory dziennika zdarzeń, 183
 - interfejs, 181
 - komponenty, 172
 - komunikacja między dostawcą i konsumentem, 172
 - końcówka interfejs, 180

- usługi sieciowe
 - nazwa, 176
 - przepływ danych, 174
 - SOAP a REST, 186
 - wybór, 193
 - standard opisu, 172
 - tworzenie
 - Apache CXF, 173
 - na podstawie pliku WSDL, 173
 - typu
 - REST, 20
 - RPC, 176
 - uwierzytelnianie, 251
 - wdrożenie, 177
 - dziennik zdarzeń, 177
 - wybór między wersją RPC i Document, 177
 - wygląd konsoli webowej, 178
 - zmiana stanu zasobu, 185
 - UTWORZENIE, 186
 - uwierzytelnianie, 240
 - BASIC, 252
 - deskryptor wdrożenia webowego JBoss, 244
 - dwuetapowe, 254
 - formularz, 245
 - strona logowania i błędu, 245
 - klienta, 254
 - samodzielny, 251
 - webowy, 251
 - mechanizm BASIC, 243
 - metody EJB, 249
 - ograniczenia bezpieczeństwa, 244
 - proste, 254
 - reguły, 260
 - serwera, 254
 - usługi sieciowe, 251
 - włączenie, 243
 - wzajemne, 254
 - użytkownicy
 - admin, 247
 - admin1234, 219
 - demouser, 242
 - dodawanie, 36
 - hasło, 37
 - jboss, 111
 - kontroliery hostów, 37
 - mechanizm uwierzytelniania, 36
 - nazwa, 37
 - rola, 242, 247
 - root, 23
 - tworzenie, 242
 - typ użytkownika, 36
 - zarządzający, 217
 - tworzenie, 218
- V**
- value, 98
 - Verisign, 258
- W**
- walidacja
 - dodanie walidacji ziarna, 118
 - JavaBeans Validation, 118
 - WAR, 19, 201
 - warstwa
 - dostępu do danych, 136
 - transportowa, 252
 - wdrażanie aplikacji, 39
 - EJB, 61
 - klastrowej, 220, 228
 - dodatek exec, 229
 - dziennik zdarzeń, 229
 - uruchamianie klienta, 230
 - w węzle domenowym, 229
 - wyłączanie węzła, 231
 - klienckiej, 71
 - trwałego magazynu danych, 132
 - webowej, 106
 - Web ARchive, 19
 - Web Service Description Language, 172
 - web.xml, 187, 220, 240
 - distributable, 236
 - klastry aplikacji webowych, 232
 - login-config, 245
 - security-constraints, 243, 251
 - wywoływanie stron JSF, 100
 - webapp, 131
 - webapp-javaee6, 88, 114, 161, 175
 - WebArchive, 141
 - WEB-INF, 88, 100, 240
 - webowa konsola zarządzania, 21
 - welcome-content, 34
 - Weld, 87
 - plik konfiguracyjny, 88
 - testowanie aplikacji, 140
 - wersje dodatków, 62
 - węzeł, 198
 - klastra, 213
 - serwerów aplikacyjnych, 32
 - wiadomość JMS, 159
 - widok
 - book.xhtml, 128, 130
 - JUnit, 144
 - setup.xhtml, 128
 - soapUI Logs, 179
 - soapUI Navigator, 179
 - trybu graficznego CLI, 197
 - wiersz poleceń, 195
 - adres zasobu, 198
 - automatyczne
 - połączenie, 196
 - uzupełnianie, 200
 - łączenie ze zdalnymi hostami, 196
 - operacje na zasobach, 198
 - tryb graficzny, 197
 - uruchomienie, 196
 - utworzenie nowego źródła danych, 113
 - WireShark, 262
 - właściwość komunikatu, 151
 - właściwość
 - jboss.bind.address.management, 218
 - jboss.domain.master.address, 218
 - jboss.management.htp.port, 36
 - WRITE, 58
 - write-attribute, 199
 - WSDL, 172
 - WSDL to Java Mapping, 174
 - współbieżność ziarna
 - sterowanie, 57
 - zarządzana przez ziarno, 58
 - wstrzykiwanie zależności, 121
 - wyjątek
 - sprawdzany, 61
 - wykonania, 61

wymiana informacji
 aplikacja kliencka i zdalny
 komponent EJB, 222
 wysoka dostępność
 aplikacje JSF, 235
 javax.servlet.http.HttpSession,
 235
 removeAttribute, 235
 setAttribute, 235
 wdrożenie i testowanie, 228
 zagadnienia programistyczne,
 235
 serializacja, 235

X

XML to Java Mapping, 174

Z

zabezpieczanie
 haseł, 247
 interakcji ze stronami WWW,
 254
 komponentów EJB, 249
 komunikacji
 EJB, 258
 HTTP, 256, 257
 przed ujawnieniem
 krytycznych informacji, 252
 warstwy transportowej, 252
 zakres
 konwersacji, 126
 widoku, 107
 zakresy CDI, 86
 @ApplicationScoped, 87
 @ConversationScoped, 87
 @Dependent, 87
 @RequestScoped, 87
 @SessionScoped, 87, 92
 zależności
 junit, 139
 zapytania JPA, 110
 zarządzanie
 komponentami, 83
 serwerem aplikacji, 36
 dodawanie użytkowników,
 36
 interfejs webowy, 36

wdrożeniami z poziomu
 konsoli webowej, 43
 współbieżnością, 57, 58
 wydaniami, 29
 ziarna przez JSF, 107
 zarządzanie serwerem aplikacji,
 195
 bezpośrednie API
 zarządzania, 209
 CLI, 195
 konstrukcja poleceń, 197
 tworzenie skryptów, 203
 tworzenie skryptów
 w językach
 zaawansowanych, 205
 wdrażanie aplikacji, 201
 interfejs wiersza poleceń, 195
 monitorowanie zasobów
 serwera, 210
 odczytywanie opisów
 modelu zarządzania, 209
 sterowanie serwerem, 209
 zasady bezpieczeństwa
 interfejsu, 196
 zasady bezpieczeństwa, 260
 po stronie klienta, 261
 reguły uwierzytelniania, 260
 remoting, 261
 tożsamość serwera, 260
 zasoby
 EntityManager, 121
 FacesContext, 121
 java.util.logger, 121
 zatrzymanie serwera, 25
 zdalne API dla CLI, 206
 zdalny klient, 227
 zdalny klient EJB, 64
 dodanie konfiguracji, 70
 dodanie metod
 asynchronicznych, 77
 obiekt Future, 78
 rezerwacja, 79
 sprawdzenie poczty, 80
 uruchom i zapomnij, 78
 konfiguracja pliku pom.xml,
 67
 przechowywanie haseł, 74
 reguły logowania, 70
 tworzenie
 kodu, 68
 użytkownika, 74

uruchomienie aplikacji, 71
 z poziomu Eclipse, 73
 usługa czasomierza, 75
 harmonogram wywołań
 metod zwrotnych, 76
 programowego, 75
 uwierzytelnianie klienta, 74
 zmiana portu, 71
 zestaw plików konfiguracyjnych
 XML, 36
 ziarna
 DataManager, 123
 dodanie walidacji, 118
 SeatProducer, 122
 singletonowe, 51
 sterowane komunikatami, 50
 a ziarna EJB, 157
 cykl życia, 157
 dodanie do aplikacji, 156
 konsumpcja komunikatów,
 164
 rozmiar puli instancji
 MDB, 158
 selektory komunikatów, 164
 tworzenie, 158
 zserializowanie żądania,
 157
 TicketController, 126, 130
 zarządzane przez JSF, 107
 ziarna CDI, 84, 159
 @ApplicationScoped, 87
 @ConversationScoped, 87
 @Dependent, 87
 @RequestScoped, 87
 @SessionScoped, 87
 DataManager, 122
 inna konwencja
 nazewnictwa, 86
 łatwość odwoływania, 98
 nazwa domyślna, 86
 nazwane, 85
 PollerBean, 104
 referencje do obiektu, 98
 RequestScoped, 84
 system rezerwacji biletów,
 88
 TheatreBookerBean, 91
 TheatreInfoBean, 93
 wstrzykiwanie
 do serwletu, 85
 komponentu EJB, 93

- ziarna CDI
 - zakresy, 86
 - zastosowanie, 125
- ziarna sesyjne
 - bezstanowe, 49
 - dodanie, 59
 - interfejs zdalny, 59
 - nazwa JNDI do wywołania, 64
 - usługi sieciowe, 180
 - zdalny klient EJB, 76
- przygotowanie, 58
- stanowe, 50
 - dodanie, 60
 - o wysokiej rozdzielczości, 222
 - zabezpieczenie, 250
- zmiennie
 - M2_HOME, 29
 - PATH, 30
 - sesyjne
 - money, 60, 98
 - znacznik szablonowy, 98
- związek
 - typu jeden do n, 111

Ż

- żądanie
 - ClientRequest, 191
 - CSR, 257
 - generowanie, 257
 - użytkownika, 126

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>

JBoss AS 7

Tworzenie aplikacji

JBoss to nieustannie rozwijany, popularny serwer aplikacji, wykorzystywany wszędzie tam, gdzie wymagane są najwyższa niezawodność, bezpieczeństwo i wydajność tworzonej aplikacji. Zapewnia kompletne wsparcie dla Javy Enterprise Edition (Java EE), czyli między innymi wstrzykiwanie zależności, EJB 3.1, JAX-WS czy JAX-RS. Ponadto możesz go mieć w każdej chwili za darmo! Ta książka wprowadzi Cię w jego tajniki i pokaże, jak używać go najefektywniej.

W trakcie lektury dowiesz się, jak przygotować Twój serwer do pracy, co musisz zainstalować oraz jak skonfigurować poszczególne elementy, żeby uniknąć typowych problemów. W kolejnych rozdziałach zapoznasz się z kluczowymi elementami Java EE – wstrzykiwaniem zależności (ang. Context Dependency Injection) oraz połączeniem CDI z JPA (ang. Java Persistence API). Testowanie zaawansowanych aplikacji korzystających z Java EE może stanowić nie lada wyzwanie – osobny rozdział został poświęcony projektowi Arquillian, który w znaczący sposób ułatwia to zadanie. Ponadto nauczysz się swobodnie korzystać z konsoli administracyjnej oraz łączyć serwery w klastry. Książka ta jest świetną lekturą dla wszystkich programistów Javy, korzystających z serwera aplikacji JBoss AS 7.

JBoss AS 7 zapewnia:

- pełne wsparcie dla Java EE
- najwyższą wydajność i bezpieczeństwo
- błyskawiczny start dzięki modularnej budowie
- kompletne środowisko do uruchomienia Twojej aplikacji

Poznaj moc lidera wśród serwerów aplikacyjnych!

helion.pl
księgarnia
internetowa

Nr katalogowy: 18257



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 WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-246-8664-3



9 788324 686643

Cena: 59,00 zł

Informatyka w najlepszym wydaniu