Paweł Kamiński

Spring

Wstęp do programowania aplikacji

Helion

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite/Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą AdobeStock.com

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/sprwst
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-289-0924-3

Printed in Poland.

Spis treści

Dla kogo jest ta książka?

Niniejsza książka przeznaczona jest dla osób, które są programistami, w szczególności aplikacji internetowych, a które chciałyby poznać podstawy programowania we frameworku Spring. Ideą tej pozycji jest położenie jak największego nacisku na praktyczne podejście do programowania i ograniczenie teorii do minimum niezbędnego do zrealizowania wymaganych założeń. Nie jest to też kompleksowy opis samego frameworka — celem niniejszej pozycji jest szybki i maksymalnie uproszczony wstęp do programowania w Springu — tak by Czytelnik po lekturze mógł swobodnie sam stworzyć podstawową aplikację internetową, obejmującą najczęściej występujące zagadnienia i techniki.

Mimo dość kompaktowej formy opisu Springa założeniem autora było, by książka opisywała zagadnienia w sposób wyczerpujący — podjęty temat powinien zostać opisany od początku do końca, stąd pierwsze rozdziały poświęcone są klasycznej technologii Spring, po czym następuje płynne przejście do Spring Boota.

Niestety, obszerność materiału i samej technologii Spring spowodowała, że należało wprowadzić pewne ograniczenia i wymagania. Czytelnik zasiadający do lektury powinien więc posiadać przynajmniej podstawową wiedzę z zakresu tworzenia aplikacji internetowych, idealnie przy użyciu innych dostępnych na rynku frameworków, nawet jeśli dotyczą one innych języków niż Java. Niezbędna okaże się również elementarna wiedza dotycząca obsługi relacyjnej bazy danych, a także sposobu działania protokołu HTTP czy API typu RESTful.

Zaprezentowane w książce przykłady kodów źródłowych napisane zostały w języku Java, którego składnia i opis nie zostały uwzględnione w książce, dlatego też w przypadku braku znajomości tej technologii warto najpierw zapoznać się z samym językiem, a następnie wrócić do niniejszej pozycji.

Reasumując, delikatne podstawy języka Java, baz relacyjnych i elementarnych technik tworzenia aplikacji internetowych to wszystko, co jest wymagane i zalecane przed zapoznaniem się z niniejszą pozycją. Czytelnikowi życzę, by lektura okazała się strzałem w dziesiątkę — tytułowym szybkim wstępem do świata Springa.

Konwencje zastosowane w książce

Podczas ponad 13-letniej kariery programisty, a także połowę krótszej nauczyciela akademickiego zapoznałem się z dziesiątkami książek mówiących o programowaniu. Wiele z nich to swoiste kompendia wiedzy, mające na celu maksymalne wyczerpanie danego materiału. Książki te, potężne objętościowo, opisywały każdy możliwy detal, często przekazując wiedzę w formie tabel czy też obszernych fragmentów dokumentacji danej technologii. Tak kompleksowe podejście zdecydowanie ma pewne plusy — pozwala na dogłębne zapoznanie się z treściami, zajrzenie do każdego możliwego zakątka opisywanej techniki czy narzędzia. Po przeczytaniu kilku takich pozycji zacząłem się jednak zastanawiać, czy takie podejście jest tym, czego programiści potrzebują. Czy tworzenie 700-stronnicowych książek wypełnionych w dużej mierze opisami teoretycznymi jest tym, co się naprawdę sprawdza przy nauce nowego języka czy biblioteki? Moje osobiste doświadczenia skonfrontowałem z doświadczeniami studentów i osób mi bliskich pracujących w branży IT. Wnioski nasunęły się szybko — wszyscy stwierdziliśmy, że choć teoretyczne zagadnienia i to, co się dzieje wewnątrz technologii, jest ważne, to o wiele ważniejsze jest praktyczne podejście do danego tematu.

W tym duchu postanowiłem założyć bloga, który przedstawiałby rozwiązania najczęściej spotykanych problemów i zagadnień w wybranych językach programowania. Kolejno opracowałem materiały niezbędne do prowadzenia zajęć ze studentami czy też na prowadzonych przeze mnie szkoleniach. Owe zapiski zaowocowały dwiema pozycjami książkowymi — *Laravel. Wstęp do programowania aplikacji internetowych* i *React. Wstęp do programowania* wydanymi przez wydawnictwo Helion.

W obu tych książkach moim celem było przedstawienie praktycznych zagadnień kosztem monotonnych, długich opisów, często pokrywających się z oficjalną dokumentacją danej technologii.

W podobny sposób napisana została niniejsza pozycja, gdyż skupia się ona przede wszystkim na praktyce, przekazywane przykłady mają za zadanie jak najgłębsze ukazanie procesu tworzenia aplikacji internetowych w Springu. Poszczególne rozdziały kolejno wprowadzają w świat Springa, przy tym ułożone są tak, by zazębiać się i przenikać. W zamyśle wartość prezentowanej książki miała zostać zbudowana na szybkim starcie — tak by Czytelnik zaraz po lekturze wiedział, jak zrealizować podstawową wersję dedykowanego oprogramowania, znał wszystkie klocki potrzebne do zbudowania swojej pierwszej budowli.

Książka została podzielona na wiele mniejszych objętościowo rozdziałów, tak by czytelnik mógł szybko wrócić do konkretnego zagadnienia, zajrzeć do konkretnego obszaru wiedzy bez zbędnego wertowania stron. Każdy z rozdziałów rozpoczyna się maksymalnie uproszczonym wstępem teoretycznym — uwzględniającym tylko absolutnie niezbędną wiedzę do realizacji założonych treści. Niezbędnym, kolejnym bardzo często występującym elementem jest kod źródłowy — dzięki niemu uda nam się zrozumieć cel i sposób działania programu, a jest to rzecz niezbędna do samodzielnego tworzenia oprogramowania. Wreszcie, by można było zobaczyć efekt działania kodu, niezbędne są zrzuty ekranu. Ich również jest niezwykle dużo, ale cel jest jeden — efekt naszych poczynań musi być widoczny i jednoznaczny.

Reasumując, głównymi elementy każdego rozdziału są kod, opis i wynik działania — czyli to, co najważniejsze przy wytwarzaniu oprogramowania.

ROZDZIAŁ 1.

Wstęp do frameworka Spring

Korzenie Springa sięgają października 2002 roku, kiedy to programista Rod Johnson opublikował jego pierwszą wersję, która była częścią książki *Expert One-on-One J2EE Design and Development*. Ogólną przyczyną powstania tego frameworka był brak zadowolenia z mocno skomplikowanej specyfikacji Javy Enterprise Edition w ówczesnej wersji. Co więcej, sam framework nie był konkurencją dla Javy EE, już oficjalna dokumentacja Springa mówi, że raczej rozszerzał on jej możliwości, wykorzystując przy tym jej techniki i specyfikacje, takie jak **Servlet API**, **WebSocket**, **JPA** (ang. *Java Persistence API*), **dependency injection** czy JMS (ang. *Java Message Service*).

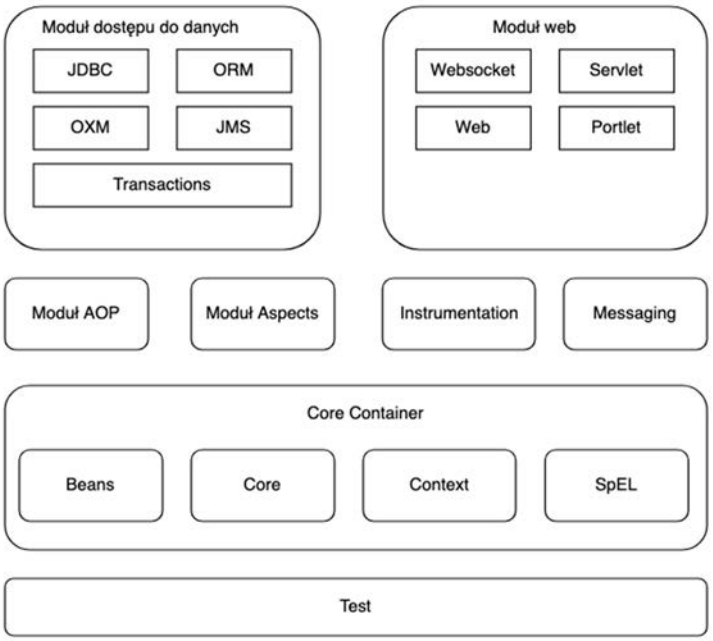
Oczywiście, przez lata sam framework ewoluował, wersja 1.0 ujrzała światło dzienne w 2004 roku, natomiast w 2006 roku powstała wersja 2.0, dodająca do dziś istniejący moduł **MVC** (ang. *Model-View-Controller*). Kolejna z wersji, 3.0 pochodząca z 2009 roku, dodała obsługę Javy w wersji 5.0 wraz z alternatywną możliwością konfiguracji — obok znanego już XML-a (ang. *Extensible Markup Language*). W wersji 4.0 dodano obsługę Javy w wersji 8, ale przede wszystkim rozpoczęto projekt Spring Boot. W roku 2017 zaprezentowano Spring 5.0, który dodał obsługę Javy 10, a także nowoczesnego języka Kotlin. Wreszcie w 2022 roku Spring został zaktualizowany do wersji 6, która do dziś jest uważaną za aktywną wersję rozwojową.

Framework Spring zawdzięcza popularność stabilnemu rozwojowi, dużym możliwościom oraz zwiększonej wydajności wytwarzania oprogramowania. Sam framework składa się z wielu modułów, idealnie dopasowanych do potrzeb współczesnych aplikacji internetowych. Wśród opisywanych modułów wyróżnić można:

- *Spring Core Container* — stanowi podstawowy element Springa, zawiera implementację wzorca IoC (ang. *Inversion of Control*) w postaci *Dependency Injection*, zawiera komponenty takie jak obsługa beanów, kontekstu aplikacji (ang. *ApplicationContext*) czy SEL (ang. *Spring Expression Language*).
- Programowanie zorientowane aspektowo — moduł pozwalający na tworzenie kodu w paradygmacie aspektowym.
- *ORM* (ang. *Object-Relational Mapping*) — poprzez opisywany moduł Spring wspiera integrację z Java Persistance API oraz z narzędziem o nazwie *Hibernate*.
- *Transaction* — moduł wspierający obsługę transakcyjności.
- *JDBC* — moduł do natywnych zapytań do bazy.
- *Spring Web-Servlet* — moduł, którego główną częścią jest implementacja wzorca MVC (ang. *Model-View-Controller*).
- *Spring Web* — moduł wspierający tworzenie stron internetowych.

- *Spring Security* — moduł wspierający proces autentykacji.
- *Spring Test* — moduł ułatwiający testowanie aplikacji; umożliwia wykorzystanie szerokiej gamy technologii i narzędzi testujących — *Junit*, *Mockito* czy też *MockMVC*.
- *Instrumentation* — moduł umożliwiający ładowanie klas w trakcie wykonywania programu.
- *Spring Data* — zespół technik ułatwiający organizację dostępu do różnych typów danych, w tym relacyjnych czy baz typu NoSQL. Dodatkowo zapewnia warstwę abstrakcji usprawniającą tworzenie podstawowych operacji na bazach.
- *Spring Cloud* — moduł oferujący zbiór narzędzi ułatwiających współpracę z chmurami.
- *Spring Batch* — moduł służący do przetwarzania dużej ilości danych.
- *JMS* — moduł służący do obsługi wiadomości.

Zaprezentowane moduły grupuje się, tworząc warstwy: dostępu do danych, aplikacji webowych czy głównego kontenera (ang. *Core Container*). Schemat ilustrujący opisywane warstwy zaprezentowany został na rysunku 1.1.



RYSUNEK 1.1. Schemat elementów frameworka Spring

Jak łatwo się domyślić, podstawowym modułem niezbędnym do choćby elementarnej pracy z frameworkiem Spring jest oczywiście kontener Springa (ang. *Core Container*). Możemy go rozpatrywać jako swoisty magazyn, którzy przechowuje poszczególne komponenty aplikacji. Komponenty te, nazywane bean, mogą ze sobą koegzystować i być w relacji. Za stworzenie opisywanej relacji i zarządzanie nią odpowiada kontekst aplikacji, który używa przy tym konfiguracji wykonanej przez programistów.

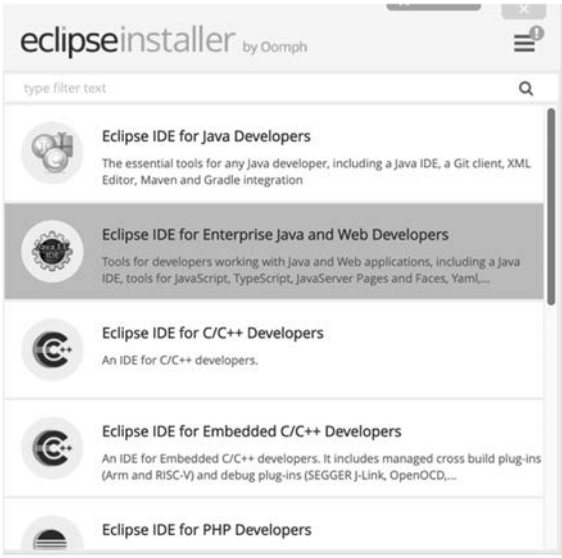
Rzeczywiste połączenie komponentów realizowane jest przy użycia wzorca *Dependency Injection*. Dzięki temu obiekty są luźno powiązane, łatwiej jest nimi zarządzać, można ich używać wielokrotnie, są łatwiej testowalne.

Pierwotnie konfiguracja beanów odbywała się poprzez odpowiednie zapisy w plikach formatu XML, dziś tych możliwości jest więcej i są one zdecydowanie bardziej przyjazne programiście — szerzej temat ten opiszę już w najbliższych rozdziałach.

Podsumowując, w dużym uproszczeniu można powiedzieć, że gdy tworzymy jeden z komponentów (bean), kontekst aplikacji zwraca nam wszystkie podległe komponenty, niezbędne do działania tego pierwotnego. Z zadań programisty wyłączone jest więc ręczne przeszukiwanie zależności pomiędzy komponentami — wszystko spoczywa na kontenerze Springa korzystającego z zaprojektowanej wcześniej konfiguracji. Do szczegółów tych zagadnień przejdziemy już wkrótce, na razie spróbujmy napisać pierwszy program korzystający ze Springa.

Pierwszą czynnością, którą powinniśmy wykonać, jest wybranie zintegrowanego środowiska programistycznego (ang. *IDE*). W książce postanowiłem użyć dwóch bardzo popularnych rozwiązań — środowiska Eclipse wraz z pakietem Spring Tools 4, a w dalszych rozdziałach — z IntelliJ IDEA. Pierwszy z nich można ściągnąć za darmo ze strony internetowej projektu Eclipse, gdzie przede wszystkim musimy wybrać rodzaj systemu operacyjnego i pobrać pliki instalacyjne.

Sama instalacja nie różni się od instalacji podobnych programów i nie powinna przysporzyć kłopotów. Na rysunku 1.2 przedstawiono opcje instalacyjne — nas interesuje wybór *Eclipse IDE for Enterprise Java and Web Developers.*

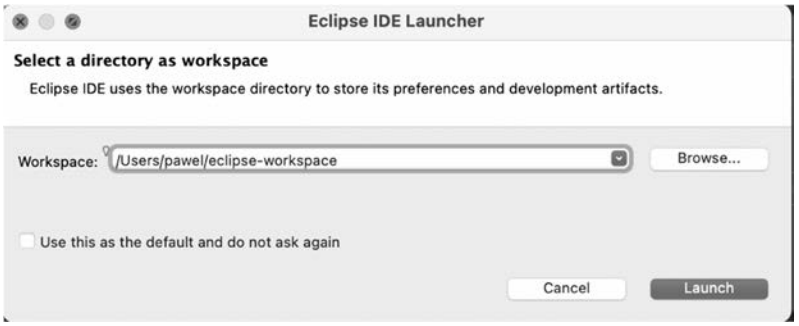


RYSUNEK 1.2. Opcje instalacyjne środowiska Eclipse

Kolejny ekran (rysunek 1.3) daje możliwość wyboru wirtualnej maszyny Javy (ang. *Java Virtual Machine*), a także miejsca, gdzie ma ona zostać pobrana.

RYSUNEK 1.3. Wybór wirtualnej maszyny Javy

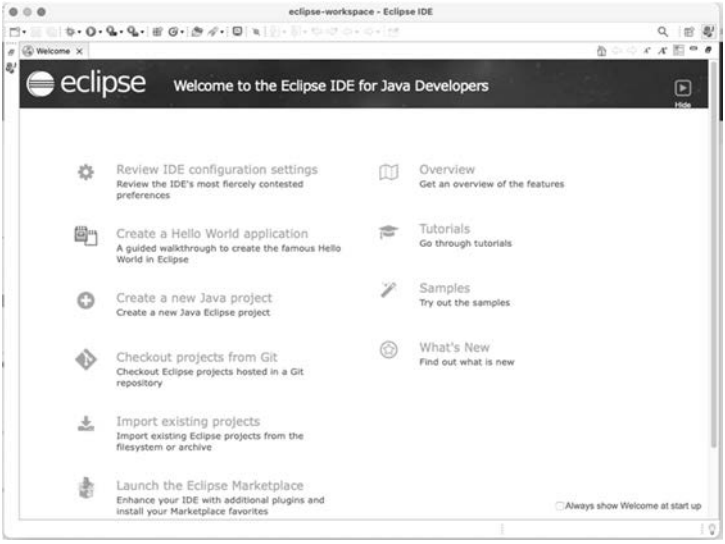
Już po uruchomieniu samego Eclipse IDE zostaniemy poproszeni o wybór miejsca w systemie plików, gdzie aplikacja będzie mogła zapisywać dane projektu (rysunek 1.4).



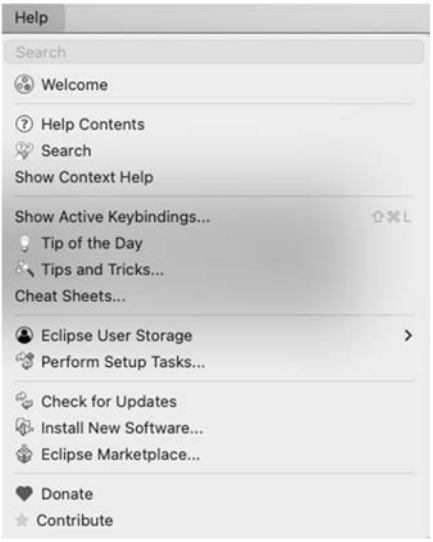
RYSUNEK 1.4. Widok wyboru folderu z projektami

Standardowe okno programu Eclipse IDE, widoczne na rysunku 1.5, składa się z typowych elementów edytora kodu. Za pomocą górnego paska narzędziowego możemy utworzyć nowy projekt, skompilować pliki źródłowe i wiele innych.

W tym momencie w celu stworzenia pierwszej aplikacji w Springu musimy zainstalować Spring Tools. W tym celu z menu *Help* wybieramy opcję *Eclipse Marketplace*, widoczną na rysunku 1.6.

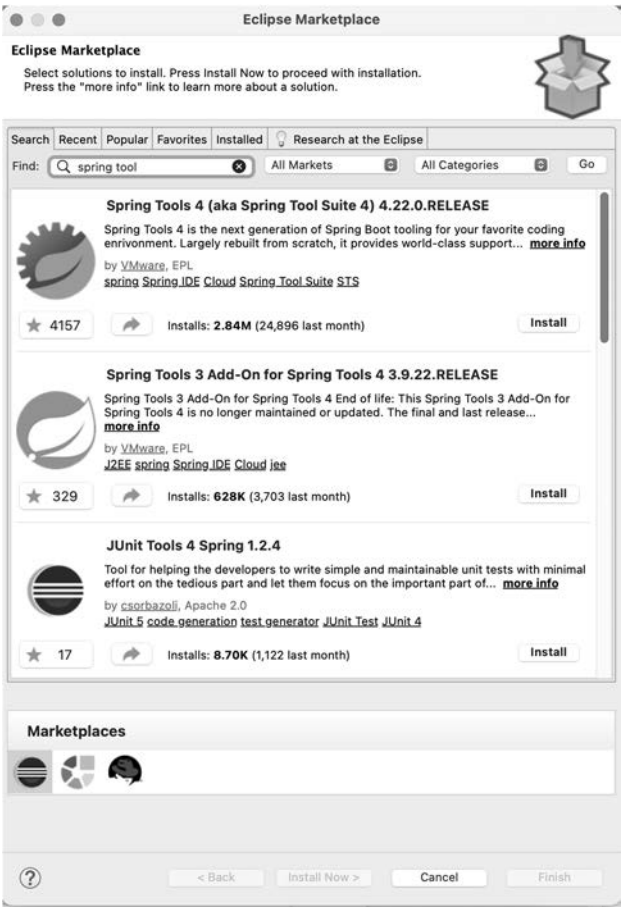


RYSUNEK 1.5. Widok okna roboczego programu Eclipse IDE



RYSUNEK 1.6. Menu Help z widoczną pozycją Eclipse Marketplace

W wyszukiwarce rozszerzeń Eclipse wpisujemy **`Spring Tools`** i wybieramy do instalacji odnaleziony moduł (rysunek 1.7).



RYSUNEK 1.7. Moduł Spring Tools widoczny w Eclipse Marketplace

Kolejne kroki instalacji to przede wszystkim wybór funkcji z listy dostępnych możliwości (rysunek 1.8), a także postanowień licencyjnych (rysunek 1.9).

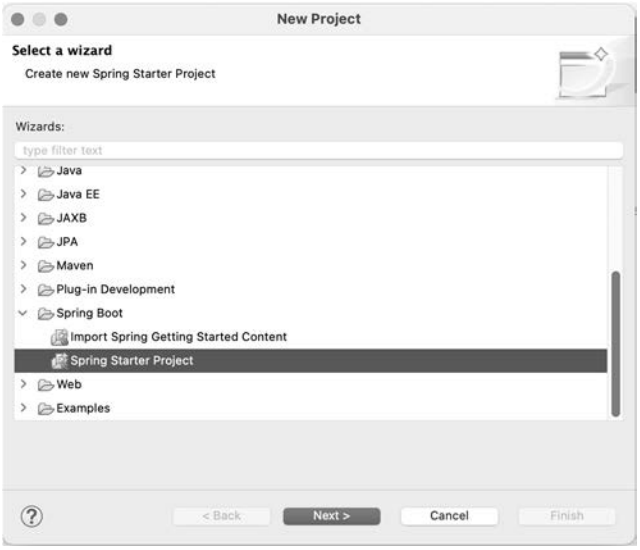
Poprawne zainstalowanie Spring Tools pozwoli nam na utworzenie nowego projektu, z listy możliwości wybieramy więc *Spring Boot* i *Spring Starter Project*. Opcja ta została przedstawiona na rysunku 1.10.

W pierwszym kroku kreatora projektu, widocznego na rysunku 1.11, możemy ustawić wszystkie szczegóły projektu, w tym:

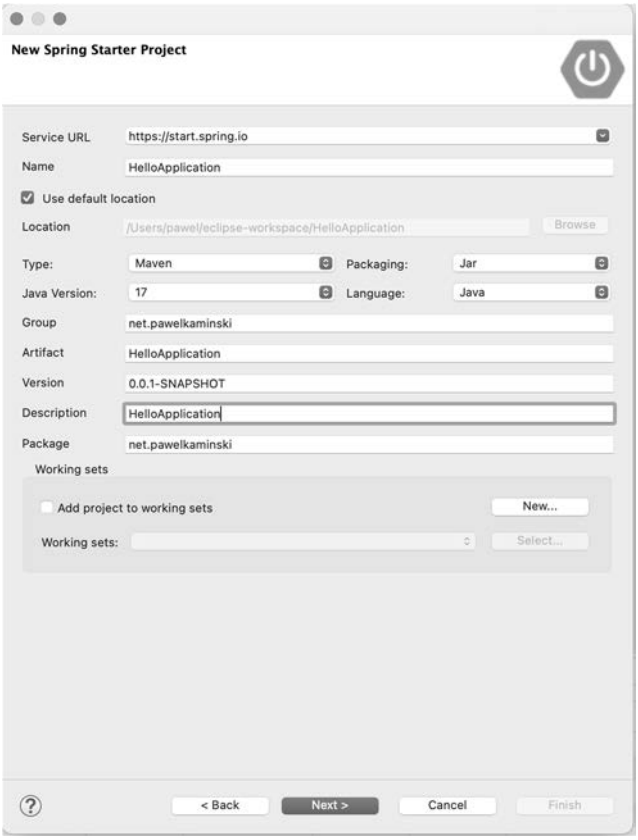
- nazwę aplikacji,
- język, w którym będziemy pisać (oczywiście Java),
- wersję języka,
- menedżera zależności (wybieramy *Maven*),
- pakiet główny aplikacji (`net.pawelkaminski`).

RYSUNEK 1.8. Wybór dodatkowych funkcjonalności

RYSUNEK 1.9. Akceptacja licencji

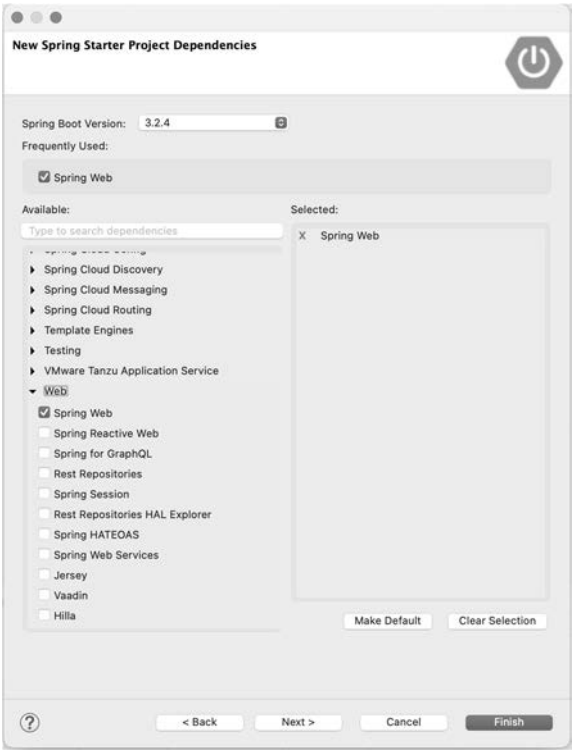


RYSUNEK 1.10. Wybór projektu Spring Starter Project



RYSUNEK 1.11. Pierwszy krok kreatora nowego projektu

W drugim kroku, zaprezentowanym na rysunku 1.12, musimy wybrać wersję Spring Boota, a także dodatkowe zależności. Z listy wybieramy kategorię *Web* i *Spring Web*.



RYSUNEK 1.12. Wybór Spring Web

Po wykonaniu powyższych kroków kreatora ujrzymy wreszcie obszar roboczy programu (rysunek 1.13), na którym widoczny będzie kod utworzony przez edytor Eclipse.

RYSUNEK 1.13. Domyślny kod projektu Spring Boot Starter

Aplikacja składa się w tej chwili z jednego pliku źródłowego *HelloApplication.java*, który zawiera definicję klasy, a także funkcję główną — `main`, uruchamiającą silnik Springa.

W celu przetestowania frameworka uzupełniamy zawartość pliku *HelloApplication .java* kodem widocznym na listingu 1.1.

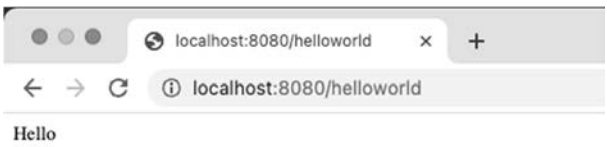
LISTING 1.1. Pierwsza aplikacja w Springu

```
package net.pawelkaminski;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
@Controller
public class HelloApplication {
    public static void main(String[] args) {
        SpringApplication.run(HelloApplication.class, args);
    }
    @GetMapping("/helloworld")
    public String hello() {
        return String.format("Hello");
    }
}
```

Kod składa się z kilku elementów:

- Adnotacji `@SpringBootApplication`, która informuje, że mamy do czynienia z klasą Springa, a także `@Controller` definiującej kontroler Springa, co umożliwia obsługę żądań HTTP. Więcej o adnotacjach będziemy mówić w kolejnych rozdziałach.
- Klasy `HelloApplication` — to nasz główny kod.
- Metody `hello`, która zwraca sformatowany napis `Hello`; metoda opisana jest za pomocą kolejnej adnotacji, tym razem jest to `@GetMapping`. Definiuje ona, że żądanie, które tworzona funkcja będzie obsługiwała, musi być typu `GET`. Dodatkowo adnotacja ta przyjmuje parametr tekstowy `/helloworld`, który precyzuje adres URI żądania, które musi zostać wywołane, by została uruchomiona metoda `hello`.

Efekt działania skompilowanego i uruchomionego kodu możemy zobaczyć na rysunku 1.14. W celu szybkiego zbudowania projektu możemy użyć skrótu klawiszowego w postaci *Ctrl+B* dla systemów rodziny Windows bądź *Shift+Command+F11* dla systemów rodziny OSX.



RYSUNEK 1.14. Efekt działania kodu wywołania metody `hello`

Rozbudowując przedstawiony przykład, możemy pokusić się o dodanie możliwości odczytu parametru paska adresu. W ramach bardzo podstawowego ćwiczenia spróbujemy pobrać z adresu URI imię osoby, którą chcemy przywitać.

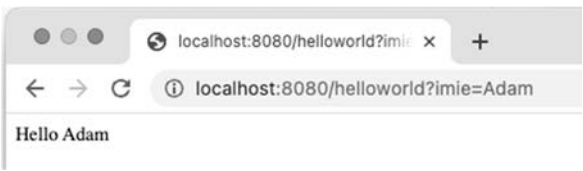
Na listingu 1.2 przedstawiono zaktualizowany kod klasy `HelloApplication`, który zawiera nową wersję metody `hello`.

LISTING 1.2. Zaktualizowana aplikacja wyświetlająca imię pobrane z parametru URI

```
package net.pawelkaminski;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.bind.annotation.RequestParam;

@SpringBootApplication
@RestController
public class HelloApplication {
    public static void main(String[] args) {
        SpringApplication.run(HelloApplication.class, args);
    }
    @GetMapping("/helloworld")
    public String hello(@RequestParam(value = "imie", defaultValue = "nieznajomy!")
String name) {
        return String.format("Hello " + name);
    }
}
```

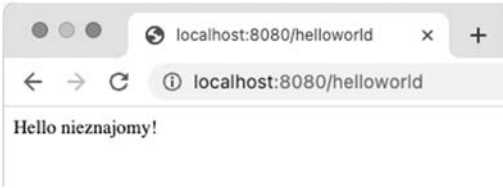
Nowa wersja metody `hello` przyjmuje parametr tekstowy `name`, który dodatkowo posiada adnotację w postaci `@RequestParam`. Adnotacja ta przyjmuje dwa argumenty — nazwę parametru pobieranego z paska adresu, a także wartość domyślną, jeśli oczekiwana wartość nie zostanie podana. Sama implementacja metody nie różni się znacząco od poprzedniej wersji — tym razem oprócz napisu `Hello` uwzględniamy również zdefiniowany wcześniej parametr.

Efekt działania kodu wraz ze zmodyfikowanym paskiem adresu uwzględniającym parametr `imie` przedstawiony został na rysunku 1.15.



RYSUNEK 1.15. Wyświetlona wartość parametru pobrana z paska adresu

W drugim przypadku użycia naszej skromnej aplikacji nie ma zdefiniowanego imienia w pasku adresu. W tym wypadku rezultat, widoczny na rysunku 1.16, obejmuje wyświetlenie domyślnego imienia `nieznajomy`.



RYSUNEK 1.16. Wyświetlenie domyślnej wartości parametru

W ten sposób udało nam się w kilku ruchach zrobić coś, co działa, odpowiada na żądania HTTP wybranego typu i wyświetla wynik na razie zdecydowanie ubogiej logiki aplikacji.

Udało nam się wykorzystać podstawowe mechanizmy Springa. Wiedzę o nich będziemy rozszerzać w kolejnych rozdziałach.

Poznaj jeden z najpopularniejszych frameworków do projektowania aplikacji w Javie

Spring należy do rodziny frameworków Javy. Jego pierwsza edycja ujrzała światło dzienne w 2002 roku i od razu spotkała się z pozytywnym odbiorem programistów. Spring Framework zyskał uznanie i popularność, ponieważ działa na zasadzie lekkiego szablonu, umożliwiającego dużą dowolność, jeśli chodzi o wybór modelu programowania. W efekcie za jego pomocą można tworzyć szerokie spektrum aplikacji — od niewielkich i prostych po potężne i bardzo skomplikowane.

To książka skierowana do programistów — przede wszystkim tych, którzy tworzą aplikacje internetowe i chcieliby zacząć pracować ze Spring Frameworkiem. Teorię ograniczono w niej do niezbędnego minimum, a główny nacisk położono na aspekty praktyczne, by Czytelnik po lekturze mógł swobodnie sam stworzyć aplikację internetową.

Z książki dowiesz się między innymi:

- **Jak zbudowany jest Spring Framework**
- **Czym się różni klasyczny Spring od Spring Boota**
- **Które elementy frameworka trzeba poznać, by zacząć projektować aplikacje**
- **Jakiego rodzaju projekty można zrealizować przy użyciu Spring Frameworka**

Paweł Kamiński

Absolwent Politechniki Białostockiej, programista z ponad 13-letnim doświadczeniem zawodowym, obecnie na stanowisku technical lead. Zajmuje się zarówno frontendem, jak i backendem. Pracował przy projektach o różnej skali — przy ugruntowanych serwisach, ale także przy aplikacjach tworzonych w startupach. Jest nauczycielem akademickim w Akademii Łomżyńskiej, gdzie był promotorem kilkudziesięciu prac inżynierskich. W życiu stawia na stały rozwój. Jego hobby to wszystko, co można uznać za retro w informatyce: gry, czasopisma, ślady po rodzimych pionierach komputeryzacji. Czas wolny od pracy i okołozawodowych pasji spędza z dwoma synami i żoną.

Helion

helion.pl

HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej! ▶

ISBN 978-83-289-0924-3

9 788328 909243

Cena: 99,00 zł