

# Wytwarzanie aplikacji w Javie przy pomocy Spring Framework - zagadnienia podstawowe

# Ćwiczenia

# Przygotowanie środowiska:

- 1. Zainstaluj JDK w wersji 6 u**ż**ywaj **ą**c standardowych opcji instalacyjnych
- 2. Ustaw zmienn**ą ś**rodowiskow**ą** u**ż**ytkownika JAVA\_HOME tak, by wskazywała katalog z JDK
- 3. Zainstaluj Mavena rozpakow uj **ą**c jego archiw um do wybranego katalogu np. C:\maven
- 4. Do zmiennej systemow ej PATH dodaj wskazanie na katalog <katalog\_mavena>/bin
- 5. Spraw dździałanie mavena wpisuj**ą**c w konsoli polecenie mvn
- 6. Aby przyspieszy **ć** proces budowania rozpakuj archiwum repository.zip do katalogu z danymi mavena <katalog\_domowy>/.m2
- 7. Zainstaluj Netbeans w wersji Java
- 8. Uruchom Środowisko Netbeans

# Lab 1\_00 – Konfiguracja aplikacji Spring

### Cel ćwiczenia:

Budowa aplikacji w oparciu o Springa.

Tworzenie deskryptorów i dostęp do beanów

### Kroki:

- 1. Pobierz moduł lab00.
- 2. W aplikacji są zdefiniowane dwie klasy: Person i Address. Zapewnij, że klasy te będą dostępne, jako beany w kontenerze. Dla klasy Person skorzystaj z deklaracji w deskryptorze xml. Dla klasy Address skorzystaj z adnotacji.
- 3. W klasie Lab0
  - 1. Skonfiguruj prosty kontener.
  - 2. Pobierz referencję do beanów i wyświetl ich właściwości.

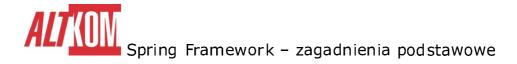
# Lab 1\_01 – Ustawianie właściwości

W poprzednim ćwiczeniu właściwości obiektów Person i Address były zaszyte w aplikacji.

### Cel ćwiczenia:

Konfiguracja właściwości beanów w deskryptorze Wstrzykiwanie zależności przez settery i konstruktor

- 1. Pobierz moduł lab01.
- 2. Zapoznaj się z konfiguracją maven
- 3. Usupełnij deskryptor context.xml o wpisy dotyczące beanów Person i Address. Wpisz własne dane, jako własności.
  - 1. Dla Address własności ustaw przez settery.
  - 2. Dla Person własności ustaw przez konstruktor.



# Lab 1\_02 – Powiązania między beanami

### Cel ćwiczenia:

Tworzenie powiązań między beanami w deskryptorze

### Kroki:

- 1. Pobierz moduł lab02.
- 2. Do klasy Entry wstrzyknij zależne obiekty klas Person, Address i Phone.
  - 1. Obiekt klasy Person wstrzyknij przez seter, jako beana zewnętrznego
  - 2. Obiekt klasy Address wstrzyknij przez seter, jako beana wewnętrznego.
  - 3. Wstrzyknięcie obiektu klasy Phone zapewnij przez zastosowanie odpowiednio adnotacji @ Autowired

# Lab 1\_03 – Wstrzykiwanie kolekcji

### Cel ćwiczenia:

Wstrzykiwanie do beana zależności, które są kolekcjami

- 1. Pobierz moduł lab03.
- 2. W desktyptorze przypisz do repozytorium pierwszego wpisy entry1, entry2, a do repozytorium drugiego entry3, entry4.
- 3. Wpisy w repozytorium pierwszym wstrzyknij jako listę, natomiast wpisy w repozytorium drugim jako mapę.

# Lab 1\_04 - Postprocessing

### Cel ćwiczenia:

Modyfikacja właściwości beanów poprzez wykorzystanie mechanizmu postprocessing

### Kroki:

- 1. Pobierz moduł lab04.
- 2. Uzupełnij implementację klasy PhonePostProcessor tak, by działała jako PostProcesor. Dla wszystkich beanów posiadających właściwości o nazwie phone i typie Phone wyświetl na konsoli telefon w postaci +<kod kraju> (<kierunkowy>) xxx-xx-xx.
- 3. Zapewnij wykorzystanie PostProcesora przez kontener.

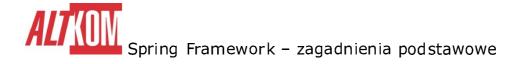
# Lab 1\_05 – Własny edytor właściwości

W poprzednim ćwiczeniu beany adresu i osoby były deklarowane jako beany wewnętrzne, w tym ćwiczeniu wykorzystamy możliwość konwersji Stringa na docelowe typy danych Address i Person

### Cel ćwiczenia:

Wstrzykiwanie do beana zależności, które powinny automatycznie skonwertowane do obiektu

- 1. Pobierz moduł lab05.
- 2. Zaimplementuj metody setAsText klas AddressEditor i PersonEditor. W implementacji zwróć uwagę na wpisy deskryptorze.
- 3. Zdefiniuj i zarejestruj edytory właściwości w kontenerze



# Lab 1\_06 – Internacjonalizacja i wiązanie beanów z kontenerem

### Cel ćwiczenia:

Uniezależnienie właściwości beanów od nazw narodowych.

Zmiana wyświetlanych tekstów w zależności od kraju i języka.

Wykorzystanie wiązania beanów z kontenerem

### Kroki:

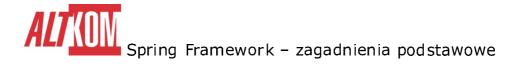
- 1. Pobierz moduł lab06.
- 2. W deskryptorze dodaj beana messageSource. Odpowiednie pliki w katalogu locale pozwolą odgadnąć wartości parametrów.
- 3. Przetestuj działanie aplikacji
- 4. Dodaj nowy język do aplikacji: zrefaktoruj odpowiednio działanie metody ComponentMediator.doLocalization()
- 5. Przenieś obiekt ComponentMediator do kontenera. AddressBookWindowBuilder powinien pobierać ten obiekt z kontenera.

# Lab 1 07 – Metody inicjalizujące

### Cel ćwiczenia:

Zastosowanie metod inicjalizujących i finalizujących.

- 1. Pobierz moduł lab07.
- 2. Do klasy InMemoryRepository dodaj metodę inicjalizującą i wykonaj w niej sprawdzenie czy pole entries zostało zainicjowane. Jeśli nie, metoda powinna rzucać wyjątek IllegalStateException.
- 3. Dodaj również metod ę finalizującą i wypisz w niej komunikat o niszczeniu klasy.
- 4. Zadanie wykonaj używając atrybutu init-method oraz adnotacji @PreDestroy.



# Lab 1\_08 – Konfiguracja aplikacji webowej

### Cel ćwiczenia:

Konfiguracji deskryptorów Spring tak, aby działały w kontekście aplikacji webowej

### Kroki:

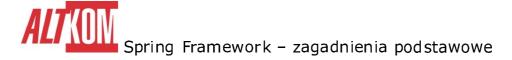
- 1. Pobierz moduł lab08
- 2. Zapoznaj się ze zmianą konfiguracji maven
- 3. Utwórz deskryptor Springa w katalogu src/main/webapp/WEB-INF
- 4. Skonfiguruj parametr kontekstu w pliku web.xml tak, aby ładowane były deskryptory Spring'a.
- 5. Skonfiguruj Servlet Dispatcher 'a tak, aby przechwytywał wszystkie URLe zakończone na \*.do
- 6. W deskryptorze Springa:
  - 1. Skonfiguruj prostego viewResolvera, aby obsługiwał JSTL.
  - 2. Skonfiguruj kontroler na link /home.do dla klasy DefaultController. Ustaw właściwości kontrolera tak, aby przekierowywał na stronę home.jsp
- 7. Wykonaj deployment aplikacji webowej.
- 8. Do przeglądarki wpisz adres: http://localhost:8080/<nazwa projektu>/home.do

# Lab 1 09 – Pobieranie danych z formularza

### Cel ćwiczenia:

Uczestnicy szkolenia będą rozwijali aplikację Książka Telefoniczna Pobranie danych od użytkownika i zapisanie ich w książce telefonicznej

- 1. Pobierz moduł lab09
- 2. Zapoznaj się ze strukturą aplikacji i elementami już zaimplementowanymi.
- 3. Stwórz kontroler o nazwie AddEntryController.
  - Celem działania kontrolera ma być: przyjęcie requestu /addEntry.do, obsługa zapisu danych wprowadzonych przez użytkownika i przekierowanie do strony



### start owej.

W implementacji posłuż się adnotacjami.

4. Rozbuduj strone addEntryForm jsp tak, aby pobrać wszystkie dane od użytkownika i zapisać je do obiektu Entry

# Lab 1\_10 – Internacjonalizacja aplikacji

### Cel ćwiczenia:

Uniezależnienie aplikacji od używanego języka

### Kroki:

- 1. Pobierz moduł lab10
- 1. Dodaj plik properties i umieść w nim wszystkie teksty używane w formularzu
- 2. Zdefiniuj obiekt messageSource
- 3. Na stronie addEntryForm.jsp odwołaj się do tekstów wpisanych w pliku properites

# Lab 1\_11 - Walidacja formularzy

### Cel ćwiczenia:

Zabezpieczenie aplikacji przed podaniem nieprawidłowych danych przez użytkownika

- 1. Pobierz moduł lab11
- 2. Napisz walidator dla formularza, który wymusi uzupełnienie wszystkich pól
- 3. Na stronie addEntryForm.jsp dodaj znacznik pozwalający wypisać błędy

# Lab 1\_12 – Konfiguracja widoku

### Cel ćwiczenia:

Wygenerowanie raportu z książki telefonicznej w formacie PDF

### Kroki:

- 1. Pobierz moduł lab12
- 2. Zdefiniuj w deskryptorze resolver odpowiedzialny za przekierowywanie obsługi żądań do beanów
- 3. Klasa EmployeesPdfReportView służy do generowania raportu w formacie PDF, umieść ją w kontenerze
- 4. Napisz kontroler, który:
  - 1. **Przechwyci żądanie <**server>/phonebook/generateReport.do
  - 2. Pobierze zawartość książki telefonicznej z repozytorium
  - 3. Umieści pobrane wpisy w modelu pod nazwą odpowiadającą nazwie widoku generującego raport PDF
  - 4. Przetestuj działanie aplikacji
  - 5. Czy potrafisz dodać proste menu, które pojawi się w momencie uruchomienia aplikacji? Menu powinno mieć możliwość wyboru dodawania wpisów albo wygenerowania raportu.

# Lab 1\_13 – Zapisywanie danych w bazie

### Cel ćwiczenia:

Zapewnienie trwałego przechowywania danych w bazie danych przy pomocy JDBC

Napisanie implementacji interfejsu PhoneBookRepository, który będzie zachowywał w bazie danych obiektu Entry

- 1. Pobierz moduł lab13
- 2. Zapoznaj się ze zmianą konfiguracji maven
- 3. Konfiguracja bazy danych:
  - 1. Przy pomocy narzędzia NETBEANS-DERBY utwórz bazę danych phonebook



## Spring Framework – zagadnienia podstawowe

- 2. W bazie danych phonebook w schemacie APP, utwórz tabelę o nazwie pb persons
- 3. W tabeli utwórz pola:
  - name typu varchar
  - surname typu varchar
- 4. W deskryptorze phonebook-repository.xml zdefiniuj źródło danych o parametrach. Parametry pobierz z pliku jdbc.properties
- 5. Zaimplementuj metody save oraz findAllEntries klasy
  JdbcRepository. Metodody powinny zapisywać obiekt Entry.person do
  tabeli pb persons.
- 6. Uruchom aplikację i przetestuj jej działanie stan bazy danych możesz podejrzeć przy pomocy narzędzia NETBEANS.

# Lab 1\_14 – Integracja aplikacji webowej z Hibernate

### Cel ćwiczenia:

Zapewnienie trwałego przechowywania danych w bazie danych, przy pomocy biblioteki Hibernate

- 1. Pobierz moduł lab14
- 2. W deskryptorze phonebook-repository.xml skonfiguruj obiekt klasy HibernateTemplate
- 3. Uzupełnij obiekt Entry o zapytania nazwane: pobierające wszystkie obiekty Entry z bazy
- 4. Napisz implementację interfejsu PhoneBookRepository wykorzystującą obiekt HibernateTempate do zapewnienia trwałego przechowywania danych
- 5. Wstrzyknij napisana implementacje do odpowiedniego kontrolera
- 6. Uruchom i przetestuj działanie aplikacji
- 7. Czy potrafisz napisać wyszukiwanie wpisów w książce telefonicznej ze względu na nazwisko?

# Lab 1\_15 – Logowanie

### Cel ćwiczenia:

Zaimplementowanie funkcji logowania za pomocą Spring AOP Kroki:

- 1. Pobierz moduł lab15
- 2. Zapoznaj się ze zmianą konfiguracji maven
- 3. W deskryptorze phonebook-service.xml włącz wsparcie dla adnotacji AspectJ
- 4. Zaimplementuj aspekt altkom.aspect.LoggerAspect
  - aspekt powinien logować wywołania metod odwołujących się do bazy danych
  - 2. logowane powinny być: nazwa klasy, nazwa metody, aktualne parametry metody oraz rezultat działania metody

# Lab 1\_16 - Transakcje

### Cel ćwiczenia:

Zapewnienie transakcyjności w aplikacji

- 1. Pobierz moduł lab16
- 2. Dodaj transakcyjność do metod pracujących bezpośrednio na źródle danych
  - 1. Skonfiguruj menadżera transakcji w deskryptorze
  - 2. Dodaj odpowiednie adnotacje do odpowiednich metod