Day 08 - Piscine Java

Spring

Резюме: Сегодня вы станете ближе к Enterprise-разработке на Java и получите представление об основах фреймворка Spring

Contents

Chapter I
Preamble

Spring Framework является неотъемлемой частью большинства корпоративных систем на Java. Данный фреймворк значительно упрощает процесс конфигурации приложения и связывания компонентов между собой. Таким образом, разработчик может полностью сосредоточиться на реализации бизнес-логики.

Принцип работы Spring полностью построен на паттернах DI/IoC, с которыми необходимо ознакомиться перед использованием данной технологии.

Центральным понятием Spring является бин (компонент), представляющий собой объект, помещенный внутри контейнера ApplicationContext. Также контейнер связывает бины между собой.

Возможно несколько способов конфигурации бинов:

1. Использование xml.файла.
2. Использование java-конфигурации (конфигурация аннотациями).
3. Совмещенная конфигурация.

XML-конфигурация позволяет изменять поведение приложения без необходимости его пересборки, в свою очередь, java-конфигурация делает код более “дружелюбным” по отношению к разработчику.

Chapter II
General Rules

- Use this page as the only reference. Do not listen to any rumors and speculations about how to prepare your solution.
- Сейчас для вас существует только одна версия Java - 1.8. Убедитесь, что на вашем компьютере установлен компилятор и интерпретатор данной версии.
- Не запрещено использовать IDE для написания исходного кода и его отладки.
- Код чаще читается, чем пишется. Внимательно изучите представленный документ с правилами оформления кода. В каждом задании обязательно придерживайтесь общепринятых стандартов Oracle - https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html
- Комментарии в исходном коде вашего решения запрещены. Они мешают восприятию.
- Pay attention to the permissions of your files and directories.
- To be assessed your solution must be in your GIT repository.
- Your solutions will be evaluated by your piscine mates.
- You should not leave in your directory any other file than those explicitly specified by the exercise instructions. It is recommended that you modify your .gitignore to avoid accidents.
- When you need to get precise output in your programs, it is forbidden to display a precalculated output instead of performing the exercise correctly.
- Have a question? Ask your neighbor on the right. Otherwise, try with your neighbor on the left.
- Your reference manual: mates / Internet / Google. И еще, для любых ваших вопросов существует ответ на Stackoverflow. Научитесь правильно их задавать.
- Read the examples carefully. They may require things that are not otherwise specified in the subject.
- And may the Force be with you!
- Не откладывайте на завтра то, что можно было сделать вчера ;)

Chapter III
Exercise 00 - Spring Context

Exercise 00: Spring Context	
Turn-in directory	ex00
Files to turn-in	Spring-folder

Реализуем слабосвязанную систему, состоящую из набора компонентов (бинов) и соответствующую принципам IoC/DI.

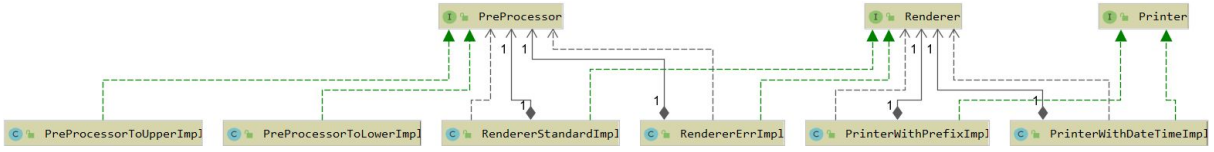
Пусть есть некоторый интерфейс `Printer`, задачей которого является вывод определенного сообщения.

Данный класс имеет две имплементации - `PrinterWithDateTimeImpl` и `PrinterWithPrefixImpl`. Первый класс выводит сообщения, указывая дату/время вывода с помощью `LocalDateTime`, второй позволяет задать для сообщения какой-либо текстовый префикс.

В свою очередь, обе имплементации `Printer` имеют зависимость на интерфейс `Renderer`, задачей которого является вывод сообщения в консоль. `Renderer` также имеет две реализации - `RendererStandardImpl` (выводит сообщение через стандартный `System.out`), и `RendererErrImpl` (выводит сообщения через `System.err`).

Также `Renderer` имеет зависимость на интерфейс `PreProcessor`, выполняющий предобработку сообщения. Реализация `PreProcessorToUpperImpl` приводит все буквы к верхнему регистру, реализация `PreProcessorToLower` - к нижнему.

UML-диаграмма классов приведена ниже:



Пример кода использования данных классов стандартным способом:

```
public class Main {
   public static void main(String[] args) {
        PreProcessor preProcessor = new PreProcessorToUpperImpl();
        Renderer renderer = new RendererErrImpl(preProcessor);
        PrinterWithPrefixImpl printer = new PrinterWithPrefixImpl(renderer);
        printer.setPrefix("Prefix");
        printer.print("Hello!");
   }
}
```

Выполнение данного кода приведет к следующему результату:

PREFIX HELLO!

Вам необходимо описать `context.xml` файл для Spring, в котором будут указаны все настройки для каждого компонента и связи между ними.

Таким образом, использование указанных компонентов с использованием Spring должно иметь следующий вид:

```
public class Main {
   public static void main(String[] args) {
       ApplicationContext context = new
ClassPathXmlApplicationContext("context.xml");
       Printer printer = context.getBean(“printerWithPrefix”, Printer.class);
       printer.print("Hello!");
   }
}
```

Chapter IV
Exercise 01 - JdbcTemplate

Exercise 01: JdbcTemplate	
Turn-in directory	ex01
Files to turn-in	Service-folder

`JdbcTemplate` и его расширение `NamedParameterJdbcTemplate` являются удобными механизмами по работе с БД. Данные классы позволяют исключить написание шаблонного кода для выполнения и обработки запросов, а также устранить необходимость перехвата проверяемых исключений.

Помимо этого, они предоставляют удобную концепцию `RowMapper`-ов для обработки `ResultSet` и конвертации результирующих таблиц в объекты.

Сейчас вам необходимо реализовать модель `User` со следующими полями:

- Идентификатор
- `Email`

Также необходимо реализовать интерфейс `CrudRepository<T>` со следующими методами:

- `Optional<T> findById(Long id)`
- `List<T> findAll()`
- `void save(T entity)`
- `void update(T entity)`
- `void delete(Long id)`

Интерфейс `UsersRepository`, объявленный как `UsersRepository extends CrudRepository<User>` должен содержать метод:

- `Optional<T> findByEmail(String email)`

Также необходимо реализовать две имплементации `UsersRepository` - `UsersRepositoryJdbcImpl` (использует стандартные механизмы `Statements`) и `UsersRepositoryJdbcTemplateImpl` (базируется на `JdbcTemaplte/NamedParameterJdbcTemaple`). Оба класса должны принимать в качестве аргумента конструктора объект `DataSource`.

В файле `context.xml` следует объявить бины на оба типа репозитория с разными идентификаторами, а также два бина типа `DataSource` - `DriverManagerDataSource` и `HikariDataSource`.

При этом, сами данные для подключения к БД должны быть указаны в файле `db.properties` и с помощью плейсхолдеров вида `${db.url}` включены в `context.xml`

Пример `db.properties`:

```
db.url=jdbc:postgresql://localhost:5432/database
db.user=postgres
db.password=qwerty007
db.driver.name=org.postgresql.Driver
```

В Main-классе необходимо продемонстрировать работу метода `findAll` с использованием обоих репозиториев:

```
ApplicationContext context = new
ClassPathXmlApplicationContext("context.xml");
UsersRepository usersRepository = context.getBean("usersRepositoryJdbc",
UsersRepository.class);
System.out.println(usersRepository.findAll());
usersRepository = context.getBean("usersRepositoryJdbcTemplate",
UsersRepository.class);
System.out.println(usersRepository.findAll());
```

Структура проекта:

- Service
    - src
        - main
            - java
                - school21.spring.service
                    - models
                        - User
                    - repositories
                        - CrudRepository
                        - UsersRepository
                        - UsersRepositoryJdbcImpl
                        - UsersRepositoryJdbcTemplateImpl
                    - application
                        - Main
            - resources
                - db.properties
                - context.xml
    - pom.xml

Chapter V
Exercise 02 - AnnotationConfig

Exercise 02: AnnotationConfig	
Turn-in directory	ex02
Files to turn-in	Service-folder

Сейчас вам необходимо настроить механизмы конфигурации Spring-приложения с помощью аннотаций. Для этого реализуйте класс-конфигурации, помеченный как `@Configuration`. Внутри данного класса необходимо описать бины для подключения к БД `DataSource` с помощью аннотации `@Bean`. Данные для подключения, как и в предыдущем задании, должны быть размещены внутри `db.properties`-файла. При этом необходимо полностью исключить наличие `context.xml`.

Также реализуйте пару интерфейс/класс `UsersService/UsersServiceImpl`, внутри которой объявлена зависимость на `UsersRepository`. Подстановка конкретного бина репозитория должна быть реализована с использованием аннотации `@Autowired` (аналогичным образом необходимо обеспечить связывание `DataSource` внутри репозиториев). Коллизии при автосвязывании следует разрешить с помощью использования аннотации `@Qualifier`.

Бины для `UsersService` и `UsersRepository` необходимо определить с помощью аннотации `@Component`.

Внутри `UsersServiceImpl` реализуйте метод `String signUp(String email)`, выполняющий регистрацию нового пользователя и сохранения информации о нем БД. Данный метод возвращает временный пароль, назначенный пользователю системой (данную информацию также необходимо сохранить в базе данных).

Для того, чтобы проверить корректность работы вашего сервиса, реализуйте интеграционный тест для `UsersServiceImpl` с использованием in-memory базы данных (`H2` или `HSQLDB`). Конфигурацию контекста для тестового окружения (`DataSource` для in-memory базы данных) необходимо описать в отдельном классе `TestApplicatoinConfig`. Данный тест должен проверять, возвращен ли какой-либо временный пароль в методе `signUp`.

Структура проекта:

- Service
  - src
    - main
      - java
        - school21.spring.service
          - config
            - ApplicationConfig
          - models
            - User
          - repositories
            - CrudRepository
            - UsersRepository
            - UsersRepositoryJdbcImpl
            - UsersRepositoryJdbcTemplateImpl
          - services
            - UsersService
            - UsersServiceImpl
          - application
            - Main
      - resources
        - db.properties
    - test
      - java
        - school21.spring.service
          - config
            - TestApplicationConfig
          - services
            - UsersServiceImplTest
  - pom.xml

CHECKLIST

https://docs.google.com/document/d/1Ps2lLpUXZD4j4FthB9lblNUpRxHhOVTl-mVL-HTa158/edit?usp=sharing