УДК 004.05

ОСНОВНЫЕ ТЕХНОЛОГИИ SPRING FRAMEWORK

Д.С. ТАТАРИН
(Представлено: канд. техн. наук, доц., доц. И.Б. БУРАЧЁНОК)

Рассмотрены особенности популярного фреймворка для разработки приложений на Java – Spring Framework, его актуальные компоненты и их возможности. Проанализированы основные преимущества применения Spring для разработчика по разработке J2EE-приложений.

Главное, что позволяет сделать Spring – упростить разработку J2EE приложений для разработчика. Поскольку J2EE-приложения в основном очень массивные и имеют долгий срок поддержки, то нужно было какое-то стандартизированное решение, которое позволяло бы перенести ответственность за инициализацию объектов и уменьшению связанности кода, что в будущем намного бы облегчало поддержку этого приложения, с разработчиков на фреймворк.

Spring Framework имеет несколько модулей, визуализацию модулей спринга можно увидеть на рисунке 1. В основе лежат модули основного контейнера, включая модель конфигурации и механизм внедрения зависимостей. Помимо этого, Spring Framework обеспечивает фундаментальную поддержку для различных архитектур приложений, включая обмен сообщениями, транзакционные данные и постоянство, а также веб. Он также включает веб-фреймворк Spring MVC на основе сервлетов и, параллельно, реактивный веб-фреймворк Spring WebFlux.

По своей сути Spring Framework представляет собой просто контейнер внедрения зависимостей, с несколькими удобными слоями (например, доступ к базе данных, прокси, аспектно-ориентированное программирование, RPC (Remote Procedure Call), веб-инфраструктура MVC (Model-View-Controller). Это все позволяет быстрее и удобнее создавать Java-приложения [1].

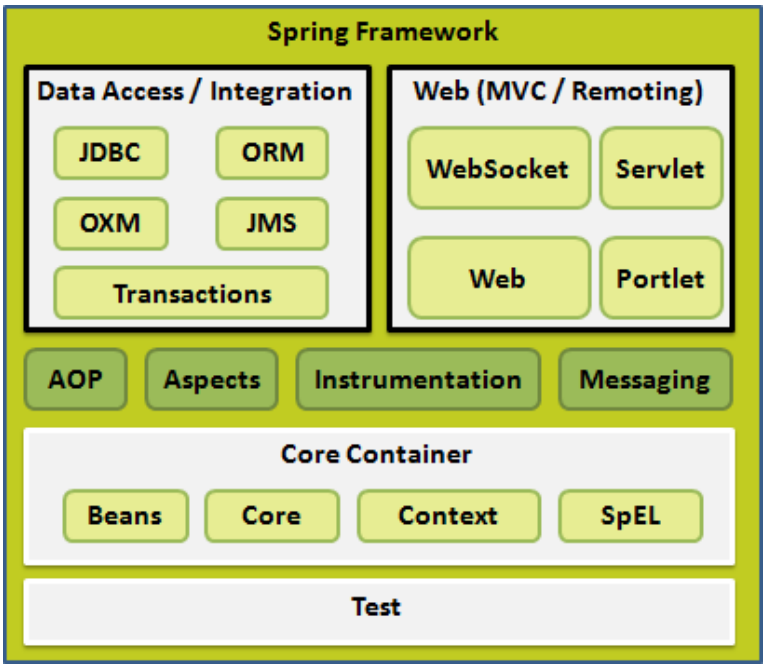

Рисунок 5. – Визуализация компонентов Spring

Далее рассмотрим основные принципы разработки, которые заложены в Spring Framework.

– Обеспечивать выбор на каждом уровне. Spring Framework позволяет откладывать дизайнерские решения как можно позже. Например, вы можете переключать поставщиков технологии программирования Object-Relational Mapping (ORM) через конфигурацию без изменения кода. То же самое верно и для многих других инфраструктурных проблем и интеграции со сторонними Application Programming Interface (API) [1].

– Учитывать разные точки зрения. Spring поддерживает гибкость и не имеет мнения о том, как все должно быть сделано. Он поддерживает широкий спектр приложений с разных точек зрения [1].

– Поддерживать обратную совместимость. Эволюция Spring Framework тщательно контролировалась, чтобы между версиями было мало критических изменений. Spring поддерживает тщательно подобранный диапазон версий Java Development Kit (JDK) и сторонних библиотек для облегчения обслуживания приложений и библиотек, зависящих от Spring [1].

– Заботится о дизайне API. Команда Spring вкладывает много времени и усилий в создание интуитивно понятных API-интерфейсов, которые выдерживают многие версии и многие годы [1].

– Установить высокие стандарты качества кода. Spring Framework уделяет большое внимание содержательной, актуальной и точной документации javadoc. Это один из очень немногих проектов, которые могут претендовать на чистую структуру кода без циклических зависимостей между пакетами [1].

Невозможно рассматривать спринг Spring Framework без понимания термина Dependency Injection (DI) – один из видов инверсии управления (Inversion of Control – IoC) [2].

При написании действительно крупных и сложных проектов разработчики сталкиваются с необходимостью делать классы приложения как можно более независимыми друг от друга для возможности повторного использования и юнит-тестирования. Именно DI устанавливает связи между этими классами, при этом сохраняя их независимость друг от друга.

Core Container (основной контейнер) включает в себя: Beans, Core, Context и SpEL (expression language). Beans отвечает за BeanFactory которая является сложной реализацией паттерна Фабрика (GoF) [1]. Бины – это объекты, которые являются основой приложения и управляются Spring IoC контейнером [1]. Эти объекты создаются с помощью конфигурационных метаданных, которые указываются в контейнере (например, XML- <bean>…</bean>). Модуль Core обеспечивает ключевые части фреймворка, включая свойства IoC и DI. Модуль Core можно найти на рисунке 1.

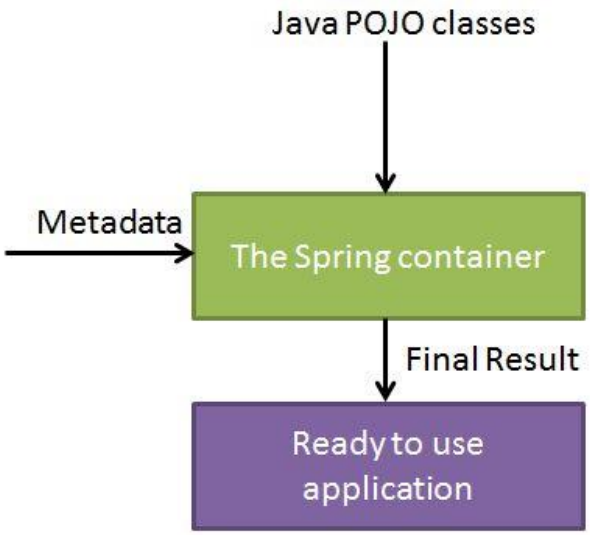

Рисунок 6. – Упрощенная работа спринга

Context построен на основе Beans и Core и позволяет получить доступ к любому объекту, который определён в настройках. Ключевым элементом модуля Context является интерфейс ApplicationContext. ApplicationContext является более сложным и более продвинутым Spring Container-ом. Так же, как BeanFactory, ApplicationContext загружает бины, связывает их вместе и конфигурирует их определённым образом. Но кроме этого, ApplicationContext обладает дополнительной функциональностью: распознание текстовых сообщений из файлов настройки и отображение событий, которые происходят в приложении различными способами. Этот контейнер определяется интерфейсом org.springframework.context.ApplicationContext.

Модуль SpEL обеспечивает мощный язык выражений для манипулирования объектами во время исполнения [2].

Контейнер Data Access/Integration состоит из: Java DataBase Connectivity (JDBC), ORM, Object/XML Mapping (OXM), Java Message Service (JMS) и модуля Transatcions.

JDBC – это стандартный API для независимого соединения языка программирования Java с различными базами данных, он обеспечивает абстрактный слой JDBC и избавляет разработчика от необходимости вручную прописывать монотонный код, связанный с соединением с БД.

ORM обеспечивает интеграцию с такими популярными ORM, как Hibernate, JDO, JPA и т.д. Модуль OXM отвечает за связь Объект/XML – XMLBeans, JAXB и т.д.

Модуль JMS (Java Messaging Service) – стандарт промежуточного программного обеспечения (ПО) для рассылки сообщений. Позволяет приложениям, выполненным на платформе Java EE, создавать, посылать, получать и читать сообщения [2].

Transactions поддерживает управление транзакциями для классов, которые реализуют определённые методы.

Веб слой состоит из Web, Web-MVC, Web-Socket, Web-Portlet. Модуль Web обеспечивает такие функции, как загрузка файлов и т. д. Web-MVC содержит реализацию Spring MVC для веб-приложений.

Web-Socket обеспечивает поддержку связи между клиентом и сервером, используя Web-Socket-ы в веб-приложениях.

Web-Portlet обеспечивает реализацию MVC в среде портлетов (элементов веб-страницы).

Spring также включает в себя ряд других важных модулей, таких как Aspect-oriented programming (AOP), Aspects, Instrumentation, Messaging и Test.

AOP реализует аспектно-ориентированное программирование (АОП) и позволяет использовать весь его арсенал возможностей.

Модуль Aspects обеспечивает интеграцию с AspectJ, которая также является мощным фреймворком АОП.

Instrumentation отвечает за поддержку Class Instrumentation и Class Loader, которые используются в серверных приложениях.

Модуль Messaging обеспечивает поддержку Simple (or Streaming) Text Oriented Message Protocol (STOMP).

Модуль Test обеспечивает тестирование с использованием TestNG или JUnit Framework.

Рассмотрев основные компоненты, можно сделать вывод, насколько Spring Framework упрощает разработку ПО, а именно берет на себя управлением зависимостями и облегчает работу с компонентами. Поэтому планируется использовать Spring Framework при реализации проекта в бэкенд части, для написания легко поддерживаемого приложения с удобным взаимодействием с архитектурой REST и базой данных.

ЛИТЕРАТУРА

1. Spring Core Technologies. Общие сведения о спринге. [Электронный ресурс] / spring. – Режим доступа: https://docs.spring.io – Дата доступа: 20.09.2020.
2. Proselyte. Руководство по спринг. [Электронный ресурс] / proselyte. – Режим доступа: https://proselyte.net/tutorials/spring-tutorial-full-version/ioc-containers – Дата доступа: 20.09.2020.