Primeiros passos com o Spring Boot

Conheça o Spring Boot, a ferramenta do ecossistema Spring que garante ganho de produtividade, qualidade e satisfação em seus novos projetos.

O que é o Spring Boot?

Há alguns anos – em abril de 2014 – a Spring IO disponibilizava a primeira versão não-beta do **Spring Boot**, a 1.0.0, depois de mais de dezoito meses de desenvolvimento e maturação.

Desde então, o projeto tem evoluído em grande velocidade e, na data de escrita deste artigo, já se encontra na versão 1.2.5, com a 1.3.x batendo a porta.

O Spring Boot é um projeto-chave para a Spring IO, mas, por ser recente, muitos desenvolvedores ainda não tiveram um primeiro contato com ele e muitos dos já introduzidos ainda desconhecem algumas ou várias de suas possibilidades e recursos.

É uma ferramenta de alavancagem de produtividade e qualidade. Assim, se você é um desenvolvedor que lida com aplicações baseadas em Spring, deve ao menos conhecê-la.

Ao ler este artigo você entenderá as motivações por trás desse projeto, assim como sua importância, não só para a Spring IO, como para a indústria de software atual.

Também irá aprender, se já não sabe, como dar seus **primeiros passos no uso do Spring Boot**, assim como orientá-lo em casos excepcionais às suas configurações pré-fixadas.

Portanto, mesmo àqueles que não estão envolvidos com o Spring em seu dia a dia, ou o fazem muito esporadicamente, este artigo trará muita informação útil, de uma forma ou de outra.

Com a finalidade de exercício dos conceitos, mecanismos e pontos apresentados, este artigo também traz um pequeno projeto web que expõe um serviço REST de consulta de aeroportos pelo mundo através dos critérios país, cidade ou código da *International Air Transport Association*, IATA. É uma aplicação um tanto mais complexa que um 'Hello World!', mas simples ao ponto de não perdermos de vista nosso foco, a **compreensão e uso do Spring Boot**.

Para construir e executá-la é preciso apenas do JDK, do Maven e estar conectado à Internet, para que o mesmo descarregue as dependências do projeto. Apesar de ser inteiramente funcional com o Java 6 e 7, replicamos aqui a recomendação do

documento de **referência do Spring Boot** em utilizá-lo com o Java 8 – como veremos, entretanto, configurá-lo a uma ou outra versão é tão simples como escrever o número dela.

Quanto ao Maven, é necessário que seja a versão 3.2+. É possível também utilizar o Gradle (1.12+), mas como o primeiro é mais conhecido e já faz parte do dia a dia da maior parte dos desenvolvedores, nosso projeto irá adotá-lo.

Apesar de código e recursos em arquivo não somarem muito mais que uma dezena de itens, recomenda-se o uso de uma IDE, preferencialmente o Spring Tool Suite (STS) – mas só pelo fato de ele possuir uma integração natural com componentes Spring, como o próprio Boot, o que facilita a execução e depuração com o mínimo esforço.

Para entendermos por que o Spring Boot foi criado, precisamos conhecer um pouco da história do Spring Framework. E para que possamos entender, de fato, o Spring Framework, precisamos conhecer um pouco da história do J2EE, Java 2 – Enterprise Edition. Comecemos, então, a discorrer sobre essas histórias e suas relações, a começar pelo J2EE. É provável que alguns dos leitores já saibam delas, mas vale a pena ver de novo.

J2EE, Java 2 – Enterprise Edition

O J2EE, parte da plataforma Java, é similarmente uma plataforma, mas específica para o desenvolvimento de aplicações Java em arquitetura de multicamadas, modular e distribuída, executadas em servidores de aplicações.

Ela não é exatamente um produto, mas um conjunto de especificações, originalmente desenvolvido pela Sun Microsystems (casa original do Java, hoje na Oracle, como sabemos), e lançado evolutivamente em versões.

A partir da versão 1.3 a especificação da plataforma passou ao domínio da JCP (*Java Community Process*), que a controla através de JSRs (*Java Specification Requests*), como a JSR-58, que especifica o J2EE 1.3 (2001), a JSR-151, que especifica o J2EE 1.4 (2002), e assim por diante.

O Java EE – passemos a utilizar a sigla atual a partir daqui – inclui diversas especificações de API, como o JDBC, e-mail, JMS, web services, RMI, JTA, JPA, JSF, JMX, JNDI, manipulação de XML, etc. e como usá-las de maneira coordenada. Há também especificações únicas a seus componentes, como EJB, servlets, portlets e diversas tecnologias de serviços web. Todas essas especificações estão interligadas com o propósito de permitir a criação de aplicações corporativas – termo comum que usamos para designar uma aplicação Java EE – portáveis entre os servidores de aplicação, os quais disponibilizam todas as especificações como infraestrutura de maneira uniforme, além de garantir escalabilidade, concorrência e gerenciamento dos componentes e aplicações entregues (deployed) neles.

O objetivo é permitir ao desenvolvedor manter o máximo possível de seu foco na construção do negócio de sua aplicação e alavancar produtividade e qualidade. Observe na **Figura 1** a visão geral do Java EE. A **Figura 2** apresenta uma visão mais detalhada e integrada.

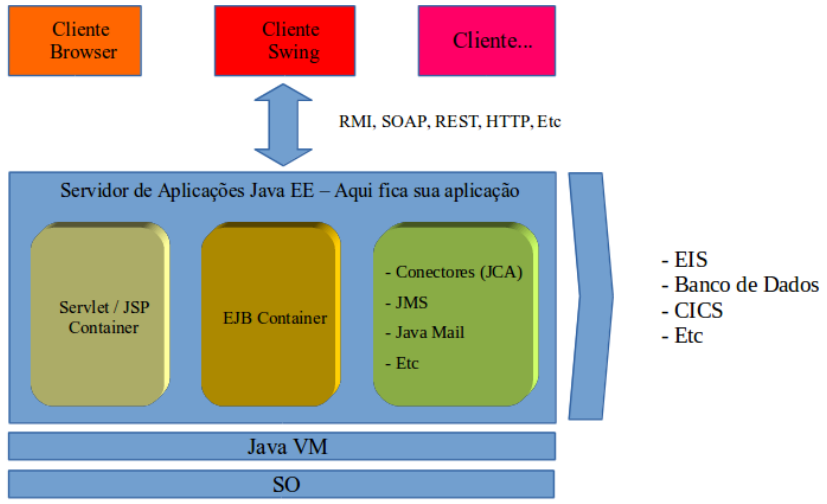

figura 2