Springboot

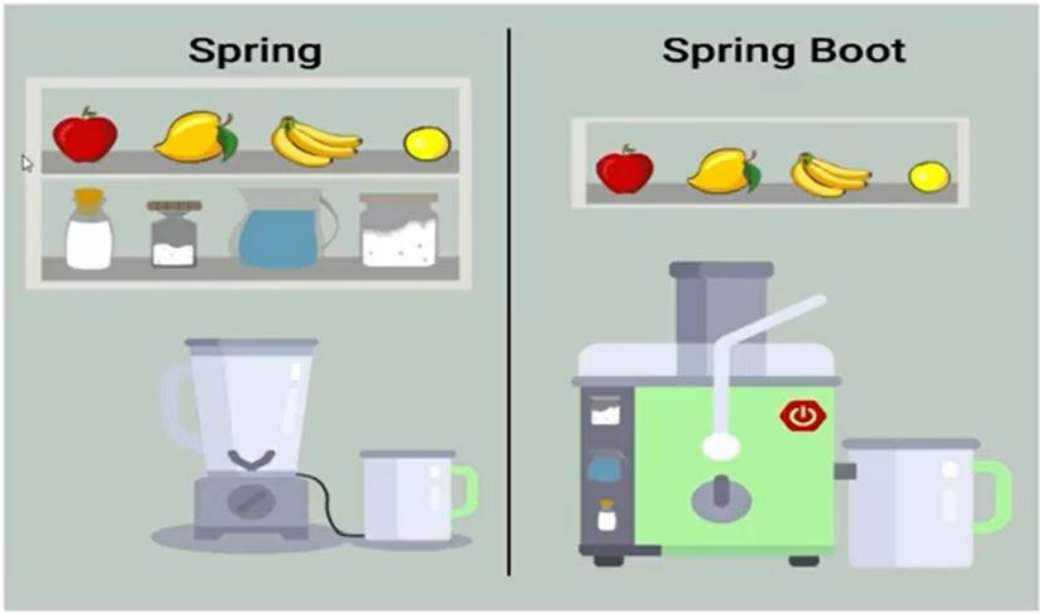
Enquanto que o Spring Framework é baseado no padrão de injeção de dependências, o Springboot foca na configuração automática.

Antes do Springboot

Desafios com a configuração do projeto.

- Dependência individual
- Verbosidade
- Incompatibilidade de versões
- Complexidade de gestão
- Configurações complexas e repetitivas

Springboot



Dado que a maior parte das configurações necessárias para o início de um projeto são sempre as mesmas, por que não iniciar um projeto com todas estas configurações já definidas?

Starters

```xml
<dependencies>
    <!-- Spring -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${springframework.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>${springframework.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
        <version>${springframework.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-orm</artifactId>
        <version>${springframework.version}</version>
    </dependency>
    <!-- Hibernate -->
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>${hibernate.version}</version>
    </dependency>
    <!-- MySQL -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>${mysql.connector.version}</version>
    </dependency>
    <!-- Joda-Time -->
    <dependency>
        <groupId>joda-time</groupId>
        <artifactId>joda-time</artifactId>
        <version>${joda-time.version}</version>
    </dependency>
    <!-- To map JodaTime with database type -->
    <dependency>
        <groupId>org.jadira.usertype</groupId>
        <artifactId>usertype.core</artifactId>
        <version>3.0.0.CR1</version>
```

Descritor de dependência

```xml
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.3.4.RELEASE</version>
    <relativePath/> <!-- lookup parent from repository -->
</parent>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <!-- MySQL -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>${mysql.connector.version}</version>
    </dependency>
</dependencies>
```

Benefícios

- Coesão
- Versões compatíveis
- Otimização do tempo
- Configuração simples
- Foco no negócio

Alguns Starters

Listagem de alguns starters mais utilizados

Spring-boot-starter-*

- **data-jpa**: Integração ao banco de dados via JPA - Hibernate.
- **data-mongodb**: Interação com banco de dados MongoDB.
- **web**: Inclusão do container Tomcat para aplicações REST.
- **web-services**: Webservices baseados na arquitetura SOAP.
- **batch**: Implementação de JOBs de processos.
- **test**: Disponibilização de recursos para testes unitários como JUnit
- **openfeign**: Client HTTP baseado em interfaces
- **actuator**: Gerenciamento de monitoramento da aplicação.

Para saber mais

- https://www.fusion-reactor.com/blog/the-difference-between-spring-framework-vs-spring-boot/
- https://dev.to/eduwyre/settling-spring-vs-spring-boot-debate-8ek
- https://www.reply.com/solidsoft-reply/en/content/webservices-soap-and-rest-a-simple-introduction
- https://www.geeksforgeeks.org/difference-between-spring-and-spring-boot/