

Spring Boot'a Giriş



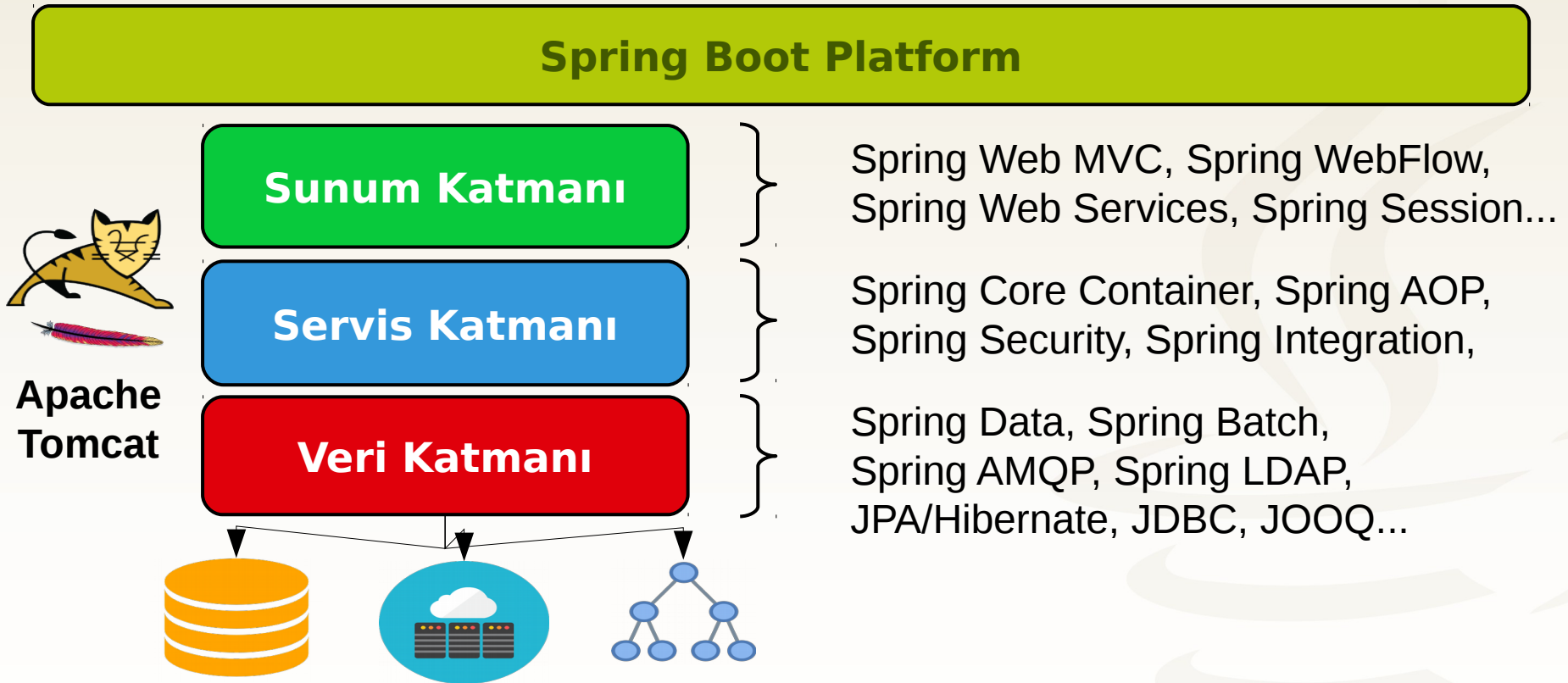
Spring Boot Nedir?

- Spring ekosistemi üzerine kurulmuş bir **framework**tür
- Temel amacı Spring ekosistemindeki teknolojiler ile **çalışmayı hızlandırmaktır**
- Kurumsal uygulamaların ihtiyaç duyduğu pek çok **altyapısal servisi** hazır biçimde sunmaya çalışır
- Uygulamaların **konfigürasyonunu ve kurulumunu** kolaylaştırır

Spring Boot Ne Değildir?

- Spring Boot bir **uygulama sunucusu veya web container değildir**, ama tomcat gibi web container'ları gömülü olarak çalıştırabilir
- **JSR spesifikasyonlarını implement etmez**, ama Hibernate gibi pek çok JSR implemantasyonunun kullanımını kolaylaştırır
- Hiçbir şekilde **otomatik kod üretmez**, konfigürasyonu otomatik hale getirir

Spring Boot'un Spring Ekosistemindeki Yeri



Spring Ekosistemi ve Spring Boot Arasındaki İlişki



**Spring Application (Core)
Framework ve diğer frameworkler...**

Spring Ekosistemi ve Spring Boot Arasındaki İlişki



Spring Boot!

Spring Boot ile Çalışmaya Başlamak

- Spring Boot ile proje geliştirirken **Maven**, **Gradle** veya **Ant** build araçlarından herhangi birisi kullanılabilir
- Maven kullanarak simple bir **Java projesi** yaratılarak çalışmaya başlanabilir
- İhtiyaç duyulan Spring Boot kabiliyetlerini aktive etmek için ilgili **bağımlılıkları kademeli olarak ekleyerek ilerlemek** mümkündür

Spring Boot ve Maven

```
<project ...>
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.javaegitimleri</groupId>
  <artifactId>petclinic</artifactId>
  <version>0.0.1-SNAPSHOT</version>
```

```
<packaging>jar</packaging>
```

JSF vb kullanan web projeleri için **war** olarak tanımlanmalıdır

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.0.0.RELEASE</version>
```

```
</parent>
```

Spring Boot'un dependency ve plugin kabiliyetlerinden yararlanmanın en kolay yolu parent pom tanımlamaktır

```
<dependencies></dependencies>
```

```
<properties>
  <java.version>1.8</java.version>
</properties>
```

Projenin ihtiyaç duyduğu kabiliyetler için Spring Boot ve diğer 3rd party bağımlılıklar buraya eklenecektir

```
...
</project>
```


Spring Boot ve Maven

```
<project...>
...
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
</project>
```



Executeable jar/war oluşturmayı sağlar
Kullanabilmek için maven 3.2 üstü ile çalışmak
gerekir

İlk Spring Boot Uygulaması

→ @SpringBootApplication, @ComponentScan ve @EnableAutoConfiguration anotasyonlarını bir araya getirir. Otomatik konfigürasyon sayesinde Spring ApplicationContext class'ındaki sınıflar ve bean tanımlarına göre otomatik oluşturulur

@SpringBootApplication

```
public class PetClinicApplication {
```

```
    public static void main(String[] args) {
```

```
        SpringApplication.run(PetClinicApplication.class, args);
```

```
    }
```

```
}
```

→ Uygulamayı JVM'de default ayarlarla çalıştırır

İlk Spring Boot Uygulaması

- Ardından IDE içerisinde **Run As Spring Boot Application** seçeneği ile çalıştırılarak <http://localhost:8080/actuator/health> adresine bir istek gönderilirse **cevap** şu şekilde olmalıdır:

```
{"status": "UP"}
```

@SpringBootConfiguration

- **@Configuration** anotasyonundan türer
- Uygulamalarda sadece bir tane **@SpringBootConfiguration** anotasyonuna sahip sınıf yer almalıdır
- **@SpringBootApplication** anotasyonu diğer kabiliyetlerin yanında bu anotasyonu da barındırdığı için genellikle o kullanılır

@EnableAutoConfiguration

- Uygulamadaki Spring **ApplicationContext**'in otomatik olarak konfigürasyonunu sağlar
- Bunun için classpath'de tespit ettiği her bir Spring Boot modülüne ait **auto-configuration sınıflarının** yüklenmesini sağlar

@EnableAutoConfiguration

- **@EnableAutoConfiguration** anotasyonuna sahip **sınıfının bulunduğu paket** özel öneme sahiptir
- Bu paket ve alt paketler uygulamaya ait **Spring bean'larını**, Spring Data Repository arayüzlerini ve **JPA entity sınıflarını** tespit etmek için taranır

@ComponentScan

- **@ComponentScan** anotasyonu ile ilave base paket'lerin de scan edilmesi sağlanabilir

```
@ComponentScan(basePackages= {"com.module1","com.module2"})
@Configuration
public class AppConfig {

}
```

↙

@SpringBootApplication anotasyonunun yer aldığı sınıfın bulunduğu paket yine base paket olarak kullanılmaya devam edilir

İletişim

- **Harezmi** Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- <http://www.java-egitimleri.com>
- info@java-egitimleri.com

