TÜBİTAK BİLGEM YTE | YAZILIM TEKNOLOJİLERİ ARAŞTIRMA ENSTİTÜSÜ

SPRING FRAMEWORK

SAYI: 05

ARAŞTIRMA SERİSİ

Simge ve Kısaltmalar

Kısaltmalar	Açıklama
TÜBİTAK	Türkiye Bilimsel ve Teknolojik Araştırma Kurumu
BİLGEM	Bilişim ve Bilgi Güvenliği İleri Teknolojiler Araştırma Merkezi
YTE	Yazılım Teknolojileri Araştırma Enstitüsü
AWS	Amazon Web Services (Amazon Web Servisleri)
API	Application Programming Interface (Uygulama Programlama Arayüzü)
AOP	Aspect Oriented Programming (Cephe Yönelimli Programlama)
MVC	Model-View-Controller (Model-Görünüm-Denetleyici)
JVM	Java Virtual Machine (Java Sanal Makinesi)
IoC	Inversion of Controller (Kontrolün Tersine Çevrilmesi)
DI	Dependency Injection (Bağımlılık Enjeksiyonu)
REST	Representational State Transfer (Temsili Durum Transferi)
HTML	Hyper Text Markup Language (Hiper Metin Transfer Protokolü)
TAS	Tanzu Application Service (Tanzu Uygulama Servisi)
PKS	Pivotal Container Service (Pivot Konteyner Servis)
FaaS	Function as a Service (Servis Olarak İşlev)

Yazar
Abdulkadir Taha YAMAÇ

Yayın Koordinatörü
Elif ŞENYİĞİT

Editörler
Hasan Çağrı TRAŞ
Sevinç KARAKAŞ
Tuğçe YILMAZ

Tasarım
Şeyma Koçer

İletişim: 0(312) 289 92 22 - yte.bilgi@tubitak.gov.tr

yte.bilgem.tubitak.gov.tr

İçindekiler

Önsöz

TÜBİTAK BİLGEM Yazılım Teknolojileri Araştırma Enstitüsü, 2012 yılından bu yana yazılım teknolojilerinde AR-GE faaliyetleri yürüten bir araştırma kuruluşudur. Araştırma faaliyetlerinde elde ettiği birikimini stratejik, hassas ve kritik projeler yürüterek kamu adına hayata geçirmekte; kurumlarımıza dijital dönüşüm, yazılım geliştirme teknolojileri ve kalite süreçleri konusunda danışmanlık vermektedir.

Araştırma Serisi ile TÜBİTAK BİLGEM YTE kurum içi çalışmaların yaygınlaştırılması ve sektörün erişimine açılması amaçlanmaktadır. Araştırma Serisi'nde yayınlanan çalışmalar TÜBİTAK BİLGEM YTE çalışanlarının projelerde elde ettiği bilgi birikimini paylaşmak adına derlenmiştir. Bu çalışmalar ile ülkemizin yazılım sektörüne katkı sağlanması hedeflenmektedir.

Giriş

Spring Framework, Java uygulamalarını geliştirmek için kapsamlı altyapı desteği sağlayan ve açık kaynak kodlu bir geliştirme platformudur. Son zamanlarda güncel teknolojiler arasında popülerliğini arttırmış ve en çok tercih edilen uygulama çerçevelerinden (framework) biri olarak öne çıkmıştır.

Spring Framework, Java dilinde uygulama geliştirirken kullanılan en önemli araçlardan biri olarak kabul edilmektedir. Bu framework, Java uygulamalarının geliştirilmesi sırasında sıkça karşılaşılan zorlukları ortadan kaldırmayı hedefler. Tüm program kodunu sıfırdan yazma çabasını ortadan kaldırarak uygulama yazma sürecini ve aşamalarını kolaylaştırır.

Açık kaynak olması, güncel ve büyük topluluklara sahip olması, ihtiyaçları karşılama noktasında yeterliliği ve kullanım kolaylığı nedenleriyle hala tercih edilen bir framework olarak öne çıkmaktadır. Spring Framework, özellikle büyük ölçekli projelerin geliştirilmesi sırasında kullanılan bir araçtır. Birçok büyük şirket, Spring Framework ile kendi projelerini geliştirmektedir.

Spring Framework genel kavramları arasında, Inversion of Control (IoC) ve Dependency Injection (DI) yer alır. Bu kavramlar, Java uygulamalarının geliştirilmesi sırasında büyük önem taşımaktadır. Bunların yanı sıra Spring Framework, AOP (Aspect-Oriented Programming), JDBC (Java Database Connectivity), ORM (Object-Relational Mapping), MVC (Model-View-Controller) gibi diğer konularda da kapsamlı bir destek sunar.

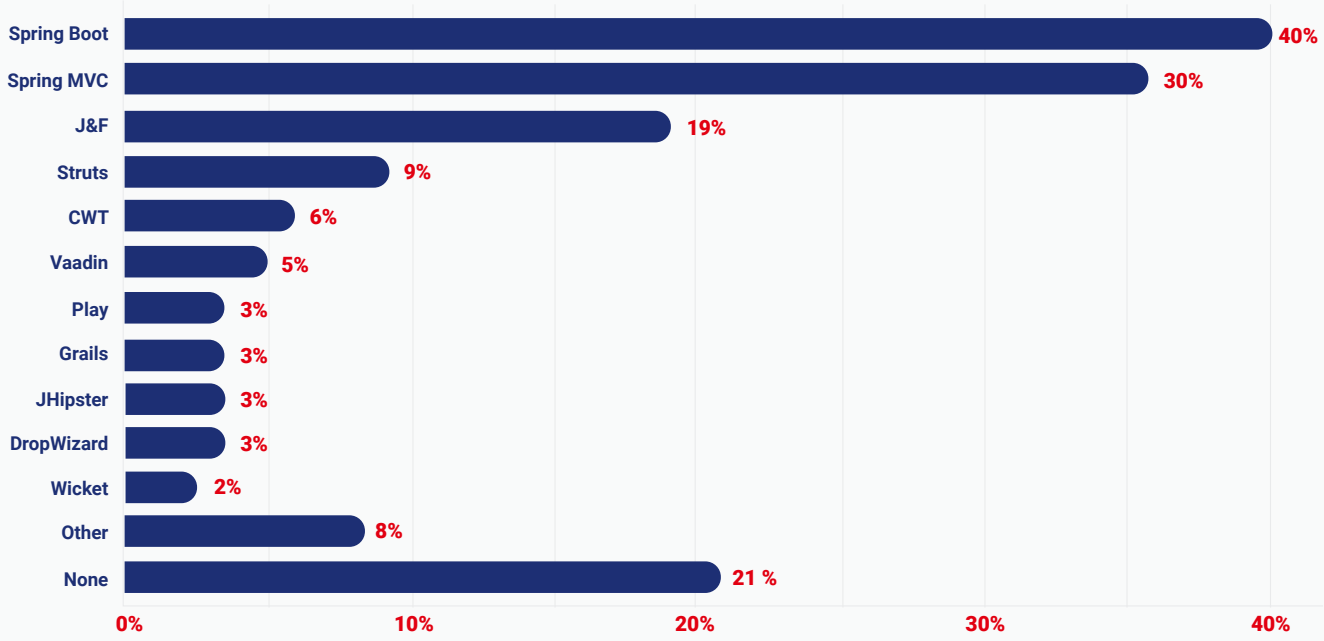
Spring Framework, alternatifleri arasında en uygun ve doğru çözümleri sunarak açık ara önde yer alır. Yapılan incelemelerde de Spring Framework rakiplerine göre daha iyi performans göstermektedir. Bu nedenle, Java uygulamalarının geliştirilmesi sırasında, Spring Framework kullanmak, projenin başarıya ulaşması için önemli bir adım olarak kabul edilmektedir.

Bu çalışma kapsamında genel olarak Spring Framework kavramları, özellikleri, kullanılabileceği alanlar, neler yapılabileceği, tercih edilme durumları gibi konulara değinilmiştir.

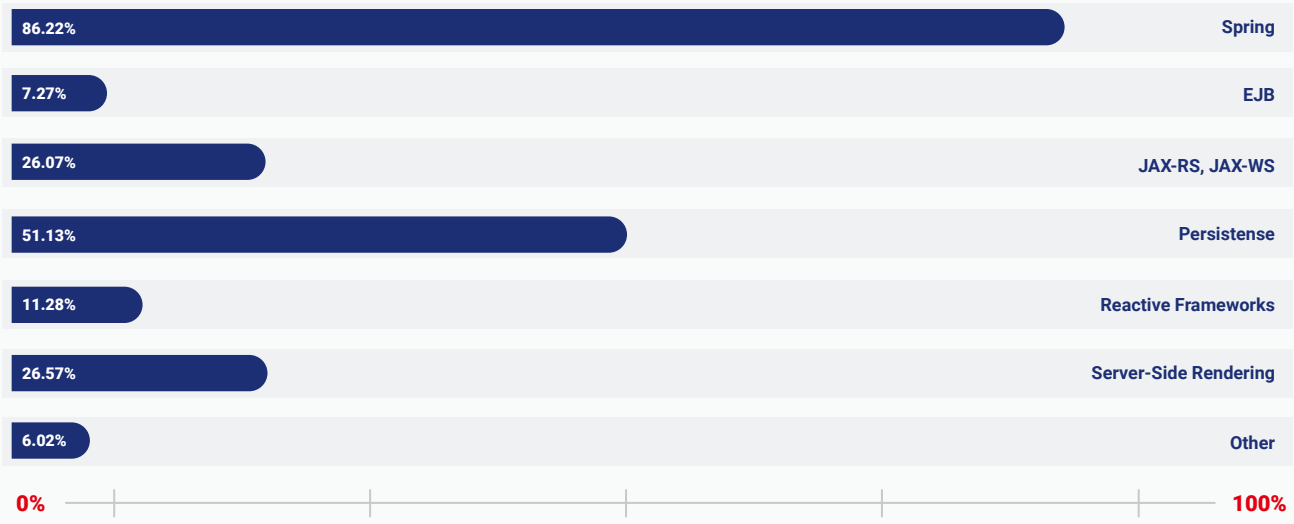
Spring Framework

Spring Framework, Java programlama dili için açık kaynak kodlu bir uygulama geliştirme platformudur. İlk olarak 2002 yılında Rod Johnson tarafından geliştirilen Spring, günümüzde birçok geliştiricinin tercih ettiği popüler bir framework haline gelmiştir.

Java teknolojileri hakkındaki birçok anket raporuna göre en çok kullanılan framework Spring olarak karşımıza çıkmaktadır. Örnek olarak Şekil 1 ve Şekil 2'de bulunan 2018 ve 2020 yıllarında yapılmış anket sonuçlarına bakılabilir;



Şekil 1. En Çok Tercih Edilen Framework Anketi 2018

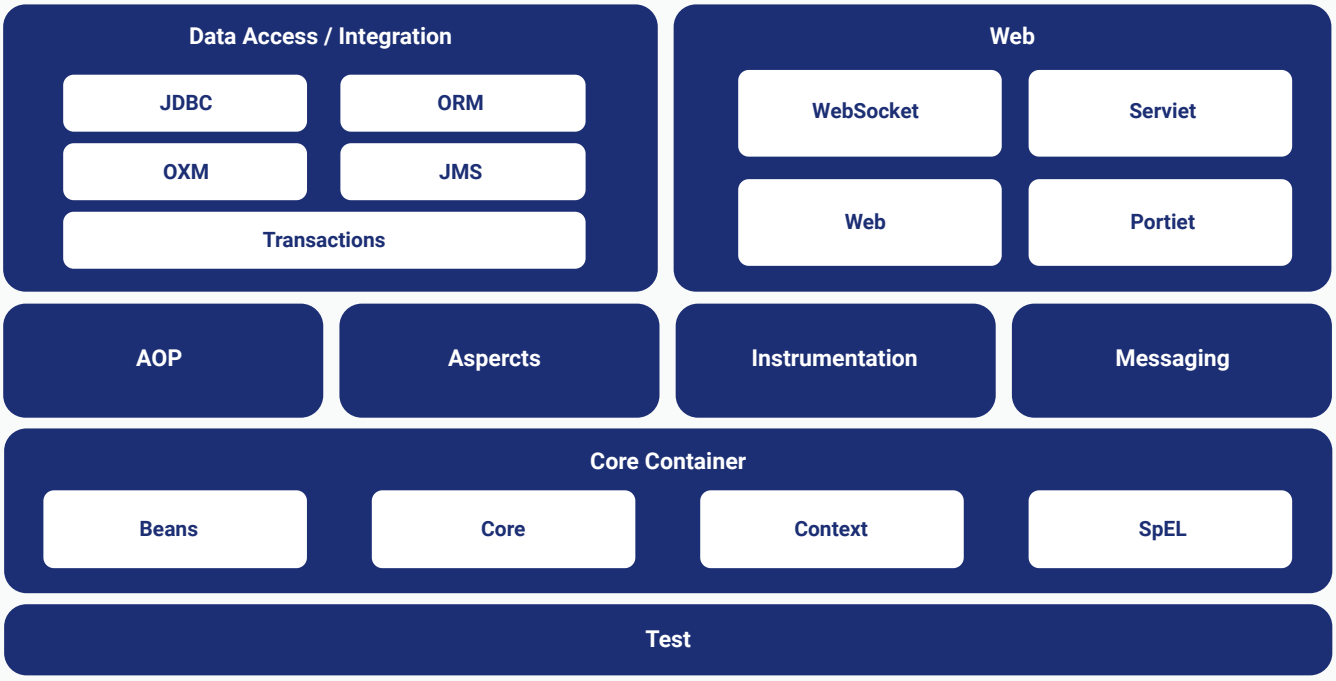


Şekil 2. En Çok Tercih Edilen Framework Anketi 2020

Spring Framework, geliştiricilere geniş bir altyapı sağlar ve hızlı bir şekilde uygulama geliştirmelerine olanak tanır. Özellikle büyük ve karmaşık projelerde kullanılmak üzere tasarlanmıştır. Yazılım geliştirme sürecini kolaylaştırmak için birçok araç ve özellik içerir. Uygulama yazma sürecinde tekrar kullanılabilir kod parçalarını destekler. Bu sayede geliştiriciler, mevcut kodları tekrar kullanarak uygulama geliştirme sürecini hızlandırabilirler. Ayrıca, yazılım tasarım kalıplarını da destekler ve bu kalıpların kullanımını kolaylaştırır.

Açık kaynak kodlu olduğu için birçok geliştiricinin katkı sağlamasına olanak tanır. Böylece, hatalar daha hızlı tespit edilir ve giderilir; yeni özellikler eklenir ve güncellemeler yayınlanır. Birçok üçüncü taraf kütüphanesi ile de entegre çalışabilir.

Spring Framework, temelde 20'den fazla modülden oluşur. Bu modüller, Çekirdek Konteyner (Core Container), Data Erişimi/Entegrasyon (Data Access/Integration), Web, AOP, Görüş (Aspects), Enstrümantasyon (Instrumentation), Mesajlama (Messaging) ve Test olmak üzere 8 ana grupta toplanır. Uygulama geliştirirken ihtiyaç duyulan modüller seçilerek kolayca bir araya getirilebilir. Şekil 3'te bu modüllerin gruplandırılması gösterilmiştir.



Şekil 3. Spring Framework Çalışma Yapısı Modeli

1. Spring Core Container

Spring Core Container dört modülden oluşur: Core, Beans, Context ve Expression Language.

Core; Spring Framework, uygulamalara entegre edilecek temel mekanizmaları içeren bir parçadır. Çalışma prensibi Inversion of Control (IoC) ilkesine dayanır. Bu prensipte, uygulamanın işleyişi kontrol etmesine izin vermek yerine, kontrol Spring Framework üzerinde konfigürasyonlar ile tanımlanır.

Inversion of Control "Inversion" terimi, uygulamanın bağımlılıkları kullanarak kendi koduyla işleyişi kontrol etmesine izin verilmemesinden kaynaklanır. Bunun yerine, çerçeve (bağımlılık), uygulama ve kodun kontrolünü ele alır (Şekil 4).

Şekil 4. IoC ile Framework Uygulamaları

IoC senaryolarında, kendi kodunu kontrol etmek yerine, uygulama bağımlılık tarafından kontrol edilir. Spring Framework, uygulamanın yürütme sırasında kontrolünü ele alır ve IoC yürütme senaryosunu uygular.

Beans; Spring Beans modülü tarafından sağlanan bir factory pattern uygulamasıdır. Bu, nesnelerin oluşturulması, yapılandırılması ve yönetilmesi için bir mekanizma sağlar. BeanFactory, nesnelerin yaşam döngüsünü yönetir ve ihtiyaç duyulduğunda onları oluşturur veya havuza alır. Bu sayede uygulama performansı artar ve kaynak kullanımı optimize edilir.

Context; kurulmuş ve yapılandırılmış herhangi bir nesneye erişim için bir araç sağlar. Core ve Beans modülleri tarafından sağlanan sağlam temel üzerine kuruludur. Bu modül, nesnelerin yaratılmasını, yapılandırılmasını ve yönetilmesini kolaylaştıran ek özellikler sunar. ApplicationContext arayüzü, Context modülünün odak noktasıdır ve genişletilmiş yapılandırma, olay yayını ve mesajlaşma gibi özellikleri içerir.

SpEL; Spring Expression Language kısaltmasıdır. Bu, Spring uygulamalarında nesne grafiğindeki nesnelere dinamik olarak erişim sağlamak ve manipüle etmek için kullanılan güçlü bir ifade dilidir. SpEL, XML ve Java tabanlı yapılandırma dosyalarında kullanılabilir ve nesnelerin özellikleri, metotları ve dizilerine erişim sağlar. Bu sayede kod yazımı daha kolay ve daha az tekrarlı hale gelir.

2. Data Access/Integration

Veri kalıcılığı, birçok uygulama için önemli bir konudur ve veritabanlarıyla çalışmak yaygın bir ihtiyaçtır. Spring Framework Veri Erişim (Data Access) modülü, bu ihtiyacı karşılamak için kullanılabilir. Bu modül, JDBC kullanımını, ORM framework entegrasyonu ve diğer veri erişim teknolojilerini içerir. JDBC, ORM, OXM, JMS ve Transactions modülleri de bu modülün parçalarıdır.

- JDBC kodu ihtiyacını ortadan kaldıran bir JDBC abstraction katmanı içerir.
- JPA, JDO, Hibernate ve iBatis, ORM modülü tarafından desteklenen entegrasyonlar içerir.
- JAXB, Castor, XMLBeans, JiBX ve XStream için OXM modülü, Object/XML eşleme uygulamalarını destekleyen bir soyutlama (abstraction) katmanı sağlar.
- JMS modülü, mesaj göndermek ve almak için olanaklar sunar.
- Özel interfaceleri ve tüm POJO kullanımlarını implement eden sınıflar için Transactions modülü, programatik ve bildiren (declarative) işlem yönetimi sunar.

3. Spring Web

Web uygulamaları ve web hizmetlerini geliştirmek için geniş bir araç seti sunar. Özellikle Spring MVC, standart bir servlet uygulaması geliştirmek için yaygın olarak kullanılır. Bunun yanı sıra, Web, Web-MVC, Web-Socket ve Web-Portlet modülleri gibi farklı araçlar da web uygulamalarının geliştirilmesinde kullanılabilir. Bu modüller, farklı ihtiyaçlara göre web uygulamalarının geliştirilmesini sağlar.

- Çok parçalı dosya yükleme ve servlet dinleyiciler (listener) aracılığıyla IoC konteyner kurulumu ve web odaklı bir uygulama ortamı dahil olmak üzere temel web odaklı entegrasyon özelliklerini içerir.
- Model-View-Controller (MVC) uygulaması Web-MVC modülünde bulunur.
- İstemci ve sunucu arasında WebSocket tabanlı iki yönlü iletişimi destekler.

4. Spring AOP

Yapıcı metot yakalayıcıları (construct method interceptors) ve nokta vuruşları (pointcuts) işlevleri ile cephe yönelimli programlamayı (AOP) mümkün kılar.

5. Spring Aspects

Sofistike ve iyi kurulmuş bir AOP framework olan AspectJ entegre edilerek kullanılır.

6. Spring Instrumentation

Belirli uygulama (application) sunucularında kullanım için enstrümantasyon (instrumentation) ve sınıf yükleyici (classloader) implementasyonları için destek içerir.

7. Spring Messaging

WebSocket alt protokolü olarak STOMP, uygulamalarda kullanım için mesajlaşma modülü tarafından desteklenir. WebSocket istemcilerinden gelen STOMP mesajlarını yönlendirmek ve işlemek için bir annotation programlama modeli destekler.

8. Spring Test

Birim (unit) ve entegrasyon (integration) testleri yazmak için kullanılacak geniş bir araç seti sunar. JUnit veya TestNG framework uygulamalarını mümkün kılar.

Spring Data

Spring ekosisteminde yer alan önemli bir proje olup, veritabanlarına kolayca bağlanmayı ve minimum sayıda kod satırı yazarak persistence katmanını kullanmayı sağlar. Hem SQL hem de NoSQL teknolojileriyle çalışabilen Spring Data, data persistence işlemlerini basitleştiren üst düzey bir katman oluşturur. Bu sayede, veritabanı işlemleri için tekrar eden kod yazma ihtiyacı ortadan kalkar ve geliştiriciler daha az zaman harcayarak daha verimli bir şekilde çalışabilirler. Hibernate, EclipseLink gibi JPA uygulamalarını ve MongoDB, Cassandra, Redis, Neo4j gibi popüler veritabanlarını destekleyen alt modülleri içerir.

Spring Boot

Spring ekosisteminde konfigürasyon üzerinde kural (convention over configuration) kavramını benimseyen bir çerçevedir. Bu kavram, bir çerçevenin tüm konfigürasyonlarını ayarlamak yerine varsayılan bir konfigürasyon sunarak gerektiği gibi özelleştirilebilmesini sağlar. Bu sayede, uygulama geliştiricileri daha az kod yazarak ve bilinen kuralları uygulayarak hızlı bir şekilde farklı uygulamalar geliştirebilirler. Spring Boot, varsayılan bir yapılandırmayla başlayarak, her uygulama için tüm yapılandırmaları yazmak yerine yalnızca kuraldan farklı olanları değiştirerek daha verimli bir şekilde uygulama geliştirmeyi mümkün kılar.

Spring Boot'un bazı temel özellikleri;

- Bağımsız (stand-alone) Spring uygulamaları oluşturabilmesi
- Gömülü bir web sunucusu (Tomcat, Jetty, Undertow) ile gelmesi
- Build konfigürasyonunu kolaylaştırmak için sağladığı başlatıcılar
- Otomatik konfigürasyon
- Kod üretimi (code generation) ve XML konfigürasyona ihtiyaç duymaması.

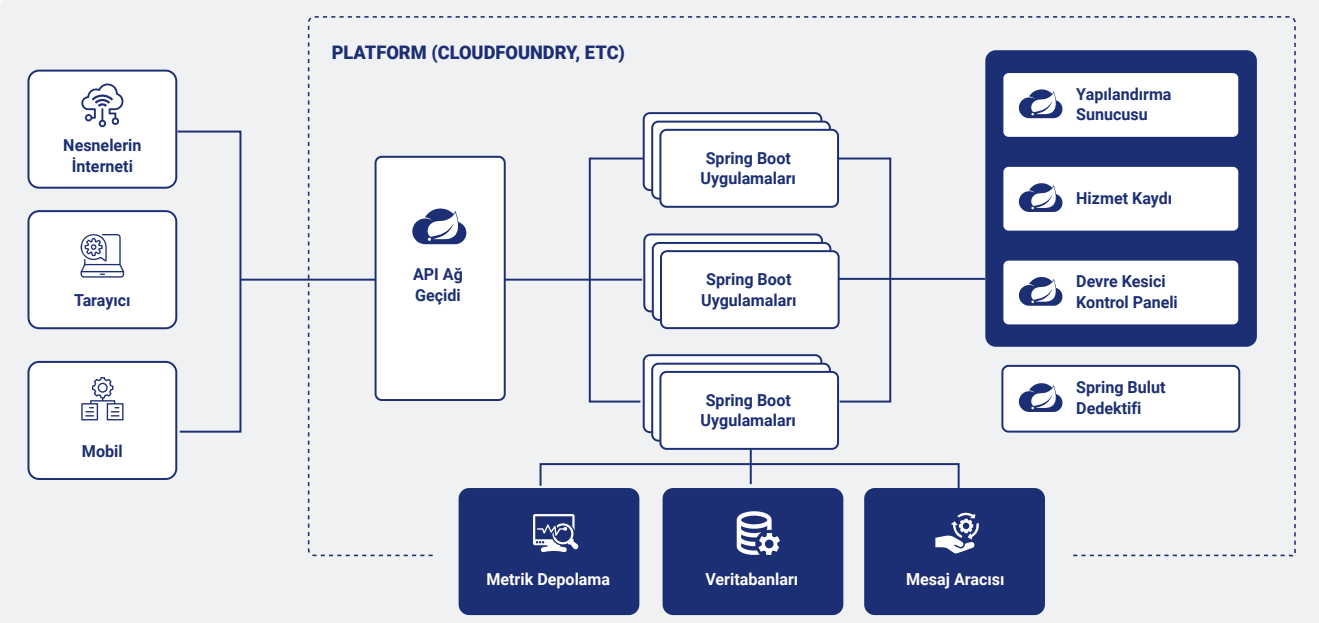
Spring ile Neler Yapılabilir?

1. Mikroservis Uygulamaları (Microservices)

Mikroservis mimarileri, günümüzde popüler olan modern bir yazılım yaklaşımıdır. Bu yaklaşımda, uygulama kodu diğerlerinden bağımsız olarak küçük, yönetilebilir parçalar halinde teslim edilir. Bu sayede; daha kolay bakım, gelişmiş üretkenlik, daha fazla hata toleransı, daha iyi iş uyumu gibi birçok ek fayda sağlanır.

Spring Boot, amaca yönelik birçok özelliği sayesinde mikroservislerinizi üretimde geniş ölçekte oluşturmayı ve çalıştırmayı kolaylaştırır. Mikroservisler, küçük ölçekli olmaları ve göreceli izolasyonları sayesinde, kodunuza büyük esneklik ve ek dayanıklılık getirebilir. Ayrıca, Spring Boot ile mikroservisleriniz küçük başlayabilir ve hızlı bir şekilde ilerlenebilir. Bu nedenle, Java mikroservisleri için fiili standart haline gelmiştir.

Spring Initializer ile projeler hızlı bir şekilde başlatılabilir ve ardından bir JAR olarak paketlenebilir. Mikroservislerin dağıtılmış doğası zorluklar getirse de, Spring bunları azaltmaya yardımcı olur. Spring Cloud, çalıştırılmaya hazır birkaç bulut modeliyle hizmet keşfi, yük dengeleme, devre kesme, dağıtılmış izleme ve izleme konularında yardımcı olabilir. Hatta bir API ağ geçidi görevi de görebilir (Şekil 5).



Şekil 5. Mikroservislerle Spring

Spring Cloud Stream, hangi mesajlaşma platformu seçilirse seçilsin, olayları tüketmeyi ve üretmeyi kolaylaştırır. Mikroservisleri, yalnızca birkaç satır kodla gerçek zamanlı mesajlaşma ile birleştirerek yüksek düzeyde ölçeklenebilir, olay güdümlü sistemler oluşturmaya yardımcı olur.

Spring Boot'un isteğe bağlı enstrümantasyon (instrumentation) aracı mikrometre (micrometer), ölçümleri doğrudan Prometheus, Atlas ve daha fazlasına göndererek değerli bilgiler sağlar. Gerçek zamanlı olarak neler olup bittiğini takip edebilmeniz için dağıtılmış izleme sunan Spring Cloud'un Sleuth ve Zipkin projeleri ile desteklenebilir.

Mikroservislerin küçük, durum bilgisi olmayan yapısı, onları yatay ölçeklendirme için ideal hale getirir. TAS ve PKS gibi platformlar, yönetim ek yükünüzü büyük ölçüde azaltacak şekilde ölçeklenebilir altyapı sağlayabilir. Bulut sağlayıcılarını kullanarak birden çok arka uç hizmetine kolaylıkla ulaşabilirsiniz.

2. Reaktif Uygulamalar (Reactive)

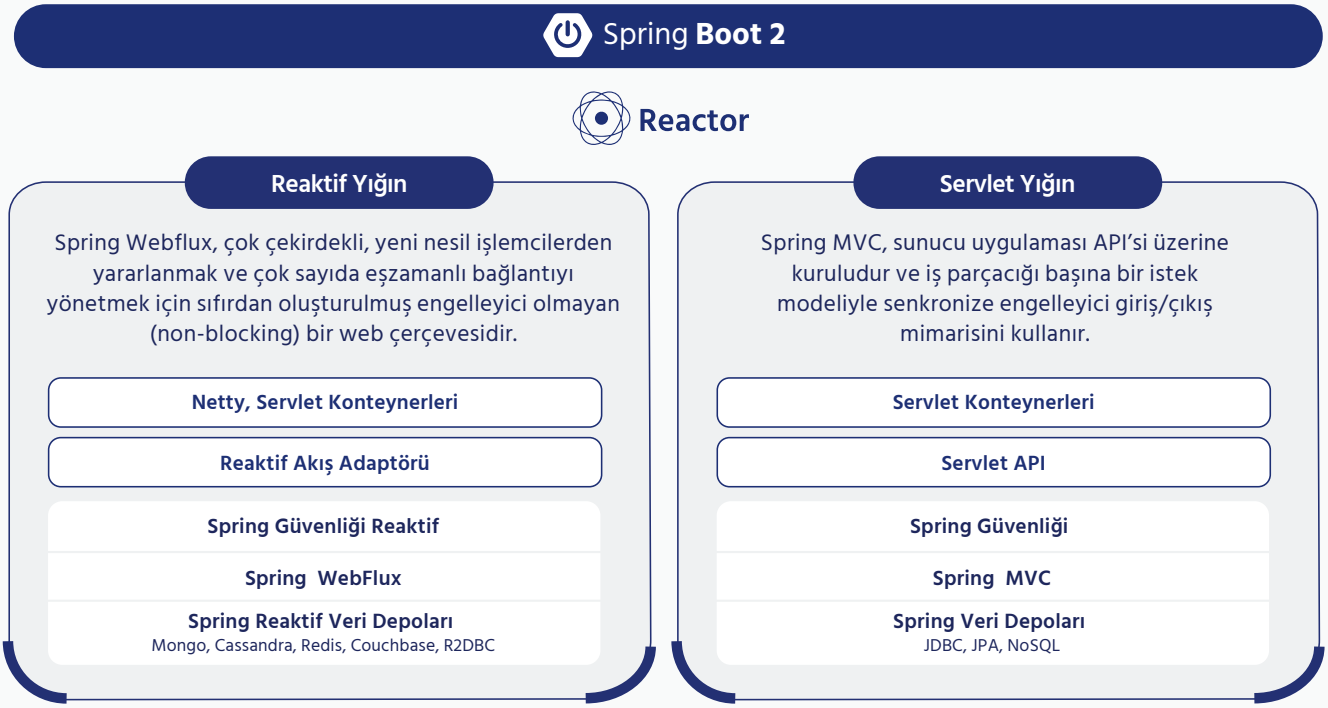
Reaktif sistemler, belirli özellikleri nedeniyle düşük gecikmeli ve yüksek verimli iş yükleri için idealdir. Project Reactor ve Spring portföyü birlikte çalışarak, geliştiricilerin duyarlı, esnek ve mesaj odaklı kurumsal düzeyde reaktif sistemler oluşturmasına olanak tanır.

Reaktif programlama, geri bildirimi (akış kontrolü) işleyebilen, engelleyici olmayan (non-blocking) ve asenkron uygulamalar oluşturmak için bir paradigmadır. Reaktif sistemler, modern işlemcileri daha iyi kullanır. Ayrıca, reaktif programlama geri bildirimin dahil edilmesiyle, ayrılmış bileşenler arasında daha iyi esneklik sağlar.

Project Reactor, tamamen engelleyici olmayan bir temel sağlayarak geri bildirim desteği sunar. Spring ekosistemindeki reaktif yığının temelidir ve Spring WebFlux, Spring Data ve Spring Cloud Gateway gibi projelerde yer alır.

Geliştiricilerin engelleyici koddan engelleyici olmayan koda geçişinin ana nedenlerinden biri verimliliktir. Reaktif kod, daha az kaynakla daha çok iş yapar. Project Reactor ve Spring WebFlux, geliştiricilerin potansiyel olarak çok sayıda eş zamanlı bağlantıyı işlemesine olanak tanır ve böylece daha az mikroservis örneğiyle daha fazla eş zamanlı kullanıcıyı memnun edebilir.

Spring portföyü, iki paralel yığın sağlar. Bir yığın, Spring MVC ve Spring Data yapılarına sahip bir Servlet API'sine dayanmaktadır. Diğer yığın, Spring WebFlux ve Spring Data'nın reaktif depolarından yararlanan tamamen reaktif bir yığındır. Her iki durumda da, Spring Security her iki yığın için de yerel destek sağlar (Şekil 6).

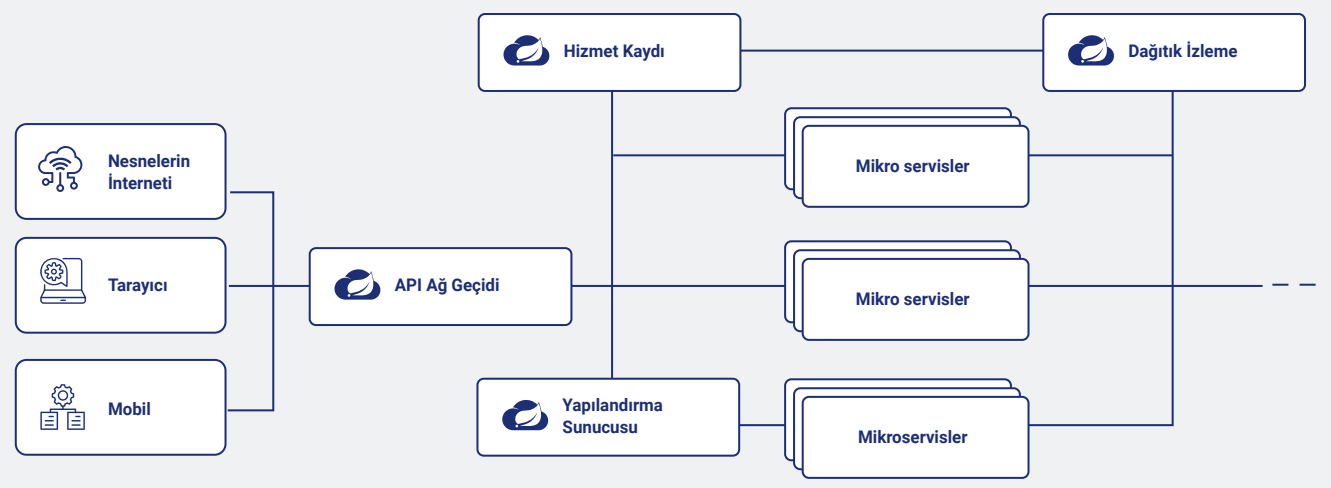


Şekil 6. Reaktif Sistemlerle Spring

Verilere reaktif bir şekilde erişmek ve işlemek önemlidir. MongoDB, Redis ve Cassandra'nın tümü Spring Data'da yerel reaktif desteğe sahiptir. Birçok ilişkisel veritabanı (Postgres, Microsoft SQL Server, MySQL, H2 ve Google Spanner) R2DBC aracılığıyla reaktif desteğe sahiptir. Mesajlaşma dünyasında Spring Cloud Stream, RabbitMQ ve Kafka gibi platformlara reaktif erişimi de destekler.

3. Bulut Uygulamaları (Cloud)

Dağıtık sistemlerin geliştirilmesi zor olabilir. Bu zorluğun sebebi karmaşıklığın uygulama katmanından ağ katmanına taşınması ve hizmetler arasındaki artan etkileşimdir. Bulut mimarisi için kodun "cloud-native" yapılması, harici yapılandırma, durumsuzluk, günlük kaydı ve destek hizmetleri gibi 12 faktörlü sorunlarla başa çıkmayı gerektirir. Spring Cloud proje paketi, bu ihtiyaç duyulan birçok hizmeti içerir (Şekil 7).



Şekil 7. Spring Bulut Mimarisi

Bulutta, uygulamalar diğer hizmetlerin tam yerini her zaman bilemez. Bunun için Netflix Eureka gibi bir hizmet kaydı veya HashiCorp Consul gibi bir sepet çözümü yardımcı olabilir. Spring Cloud, Eureka, Consul, Zookeeper ve hatta Kubernetes'in yerleşik sistemi gibi popüler kayıtlar için DiscoveryClient uygulamaları sağlar. Yükün hizmet örnekleri arasında dikkatli bir şekilde dağılmasına yardımcı olacak bir Spring Cloud Yük Dengeleyici de vardır.

Sistemde çok sayıda istemci ve sunucu varken, bulut mimarisine bir API ağ geçidi eklemek genellikle yardımcı olur. Bir ağ geçidi, mesajların güvenliğini ve yönlendirilmesini, hizmetleri gizleme, yükü azaltma ve diğer birçok yararlı şeyi halledebilir. Spring Cloud Gateway, yapılandırma ve bakımı basitleştirmek için Spring Cloud hizmet keşfini ve istemci tarafı yük dengeleme çözümlerini entegre ederek API katmanı üzerinde hassas bir kontrol sağlar.

Bulutta, yapılandırma basitçe uygulamanın içine yerleştirilemez. Yapılandırma, birden çok uygulama, ortam ve hizmet örneğiyle başa çıkacak kadar esnek olmalı ve kesinti olmadan dinamik değişikliklerle başa çıkabilmelidir. Spring Cloud Config, bu yükleri hafifletmek için tasarlanmıştır ve yapılandırmayı güvende tutmaya yardımcı olmak için Git gibi sürüm kontrol sistemleri ile entegrasyon sunar.

Dağıtık sistemler güvenilir olmayabilir. İstekler zaman aşımları ile karşılaşabilir veya tamamen başarısız olabilir. Bir devre kesici bu sorunları azaltmaya yardımcı olabilir ve Spring Cloud Devre Kesici bu konuda üç popüler seçenek sunar: Resilience4J, Sentinel veya Hystrix.

Dağıtılmış uygulamalarda hata ayıklama karmaşık olabilir ve uzun zaman alabilir. Herhangi bir arıza için, birkaç bağımsız hizmetten gelen bilgi izlerini bir araya getirmeniz gerekebilir. Spring Cloud Sleuth, uygulamaları öngörülebilir ve tekrarlanabilir bir şekilde kullanabilir. Ayrıca, Zipkin ile birlikte kullanıldığında, ortaya çıkabilecek gecikme sorunlarına odaklanabilmeye kolaylık sağlar.

Bulutta, ekstra avantajlar sağlamak için güvenilir, doğru, kararlı API'lere sahip olmak gerekir. Sözleşmeye bağlı (contract based) test, yüksek performanslı ekiplerin yolda kalmak için sıklıkla kullandığı bir tekniktir. Kodun kontrol altında kalmasını sağlamak için API'lerin içeriğini resmileştirerek ve etraflarında testler oluşturarak yardımcı olur. Spring Cloud Contract, Groovy, Java veya Kotlin'de yazılmış sözleşmelerle REST ve mesajlaşma tabanlı API'ler için sözleşme tabanlı test desteği sağlar.

4. Web Uygulamaları (Web Applications)

Spring, web uygulama geliştirme sürecini hızlı ve sorunsuz hale getirir. Ortak kod ve yapılandırma zorluklarını ortadan kaldırarak sunucu tarafı HTML uygulamaları, REST API'leri ve çift yönlü, olay tabanlı sistemler gibi modern web programlama modellerinin geliştirilmesini kolaylaştırır.

Spring Boot, geliştirme sürecinin başlangıç noktasıdır. Minimum ön yapılandırmayla, hızlı bir şekilde çalışmaya başlamayı amaçlar. Gömülü uygulama sunucuları sayesinde, saniyeler içinde hizmet vermeye başlayabilir. Ayrıca; izleme, ölçümler ve sağlık durumu gibi gelişmiş özellikleri sunarak, derinlemesine bilgi edinilmesini sağlar. Spring ayrıca; Java, Kotlin ve Groovy gibi birden çok JVM dilini destekler.

Spring Security, web uygulamalarının güvenliğini sağlama konusunda yardımcı olur. Endüstri standardı kimlik doğrulama protokollerini, SAML, OAuth ve LDAP dahil olmak üzere birçok protokolü destekler. Oturum sabitleme, tıklama hırsızlığı, siteler arası istek sahtekarlığı ve OWASP saldırıları gibi en önemli güvenlik risklerine karşı koruma sağlar.

Spring ayrıca, geliştiricilerin web uygulamalarını çeşitli veri depolarına bağlamasına yardımcı olur. İlişkisel olan ve olmayan veritabanları, harita azaltma çerçeveleri ve bulut tabanlı veri hizmetleri gibi çeşitli veri depolama seçeneklerini destekler.

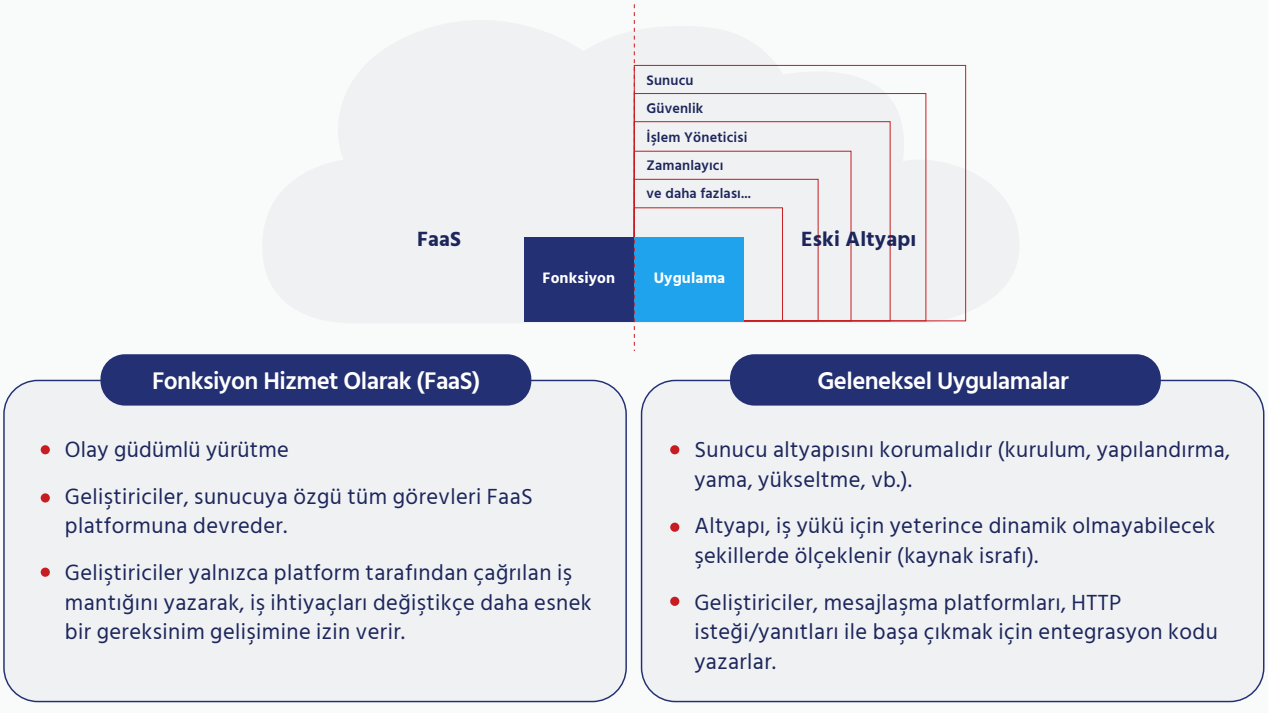
5. Sunucusuz Uygulamalar (Serverless)

Sunucusuz uygulamalar, modern bulut bilişim yeteneklerini ve soyutlamaları kullanarak altyapı yerine iş mantığına odaklanmayı mümkün kılar. Sunucusuz bir ortamda, temel platform ölçeklendirme, çalışma zamanları, kaynak tahsisi, güvenlik gibi sunucu özellikleri ile uğraşmak yerine, geliştiriciler uygulama kodlarına odaklanabilirler.

Sunucusuz iş yükleri, normalde sunucu altyapısı tarafından ele alınan yönlerle ilgilenmeyen, olaya dayalı iş yükleridir. Örneğin, kaç örnek çalıştırılacağı veya hangi işletim sisteminin kullanılacağı gibi endişeler, Hizmet olarak İşlev platformu (veya FaaS) tarafından yönetilir ve geliştiricilerin iş mantığına odaklanması için özgür bırakılır. Bu yaklaşım, geliştiricilerin daha verimli olmalarına ve iş mantığına daha çok odaklanmalarına yardımcı olur (Şekil 8).

Sunucusuz uygulamalar, aşağıdakiler de dahil olmak üzere bir dizi belirli özelliğe sahiptir:

- Olaya dayalı kod yürütme
- Başlatma, durdurma ve ölçeklendirme
- Boştayken düşük maliyete veya sıfır maliyete ölçekleme



Şekil 8. Sunucusuz ve Geleneksel Yığın Karşılaştırması

Spring portföyü, sunucusuz uygulamaların geliştirilmesi için birçok farklı araç ve kütüphane sunar. Bu araçlar, geliştiricilerin verilere erişme, entegrasyon kalıplarını kullanma ve reaktif programlama gibi konulara odaklanmalarını sağlar. Spring Data, veritabanlarına erişmek için kullanılır ve birçok farklı veritabanı teknolojisini destekler. Spring Integration, kurumsal entegrasyon kalıplarını uygulamak için kullanılır ve mesajlaşma sistemleri, e-posta ve dosya işlemleri gibi farklı entegrasyon senaryolarını destekler. Spring Framework ve Project Reactor ise, reaktif programlama için kullanılan kütüphanelerdir ve asenkron programlama modellerini desteklerler.

Spring Cloud Function ise sunucusuz uygulamalar için bir işlevsel hizmet çerçevesidir. Bu çerçeve, geliştiricilerin işlevlerini birçok farklı sunucusuz platformda çalıştırmalarını sağlar ve geliştiricilerin platforma özgü API'lerle uğraşmasını engeller. Spring Cloud Function, çekirdek Java'daki "java.util.function" paketini kullanarak işlevlerin tanımlanmasını ve yönetilmesini sağlar. Bu sayede, geliştiricilerin işlevlerini çok çeşitli sunucusuz platformlarda çalıştırmaları kolaylaşır ve platforma bağımlılıklarından kurtulmaları mümkün hale gelir. Özetle, Spring Cloud Function şunları sağlar:

- Programlama stilleri seçimi: reaktif, zorunlu veya hibrit.
- İşlev bileşimi ve uyarlama (örneğin, zorunlu işlevleri reaktif ile oluşturma).
- Fonksiyonların birleştirme ve diğer karmaşık akış işlemlerini yönetmesine izin vermek için çoklu giriş ve çıkışlarla reaktif fonksiyon desteği.
- Giriş ve çıkışların şeffaf tip dönüşümü.
- Hedef platforma özel dağıtımlar için paketleme işlevleri (Project Riff, AWS Lambda).
- Esnek imzalı işlevler (POJO işlevleri) Spring'in deyimlerinin ve programlama modelinin diğer tüm faydaları.

Spring Cloud Function, işlevlerinizi Amazon Lambda, Apache OpenWhisk, Microsoft Azure ve Project Riff dahil olmak üzere en yaygın FaaS hizmetlerinde çalıştırabilmeniz için adaptörler sağlar.

6. Olay Güdümlü Uygulamalar (Event Driven)

Olay güdümlü sistemler, modern işletmelerin gerçekte nasıl çalıştığını yansıtır. Binlerce küçük değişiklik her gün gerçekleşir ve bu değişiklikler olaylar olarak kaydedilir. Spring, bu olayları yönetme ve işlemeye yönelik bir dizi araç sağlayarak, uygulamaların işletme ile senkronize kalmasını kolaylaştırır.

Spring, olay akışı, veri akışı ve entegrasyon gibi farklı olay odaklı seçenekler sunar. Örneğin, veri akışı sabit bir olay akışını temsil eder ve hisse senedi fiyatı değiştiğinde yeni bir olay oluşur. Entegrasyon ise mesaj işleme temeline dayanır ve mesaj platformlarına bağlanma, mesajları yönlendirme, mesajları dönüştürme ve mesajları işleme gibi zorlukları çözer.

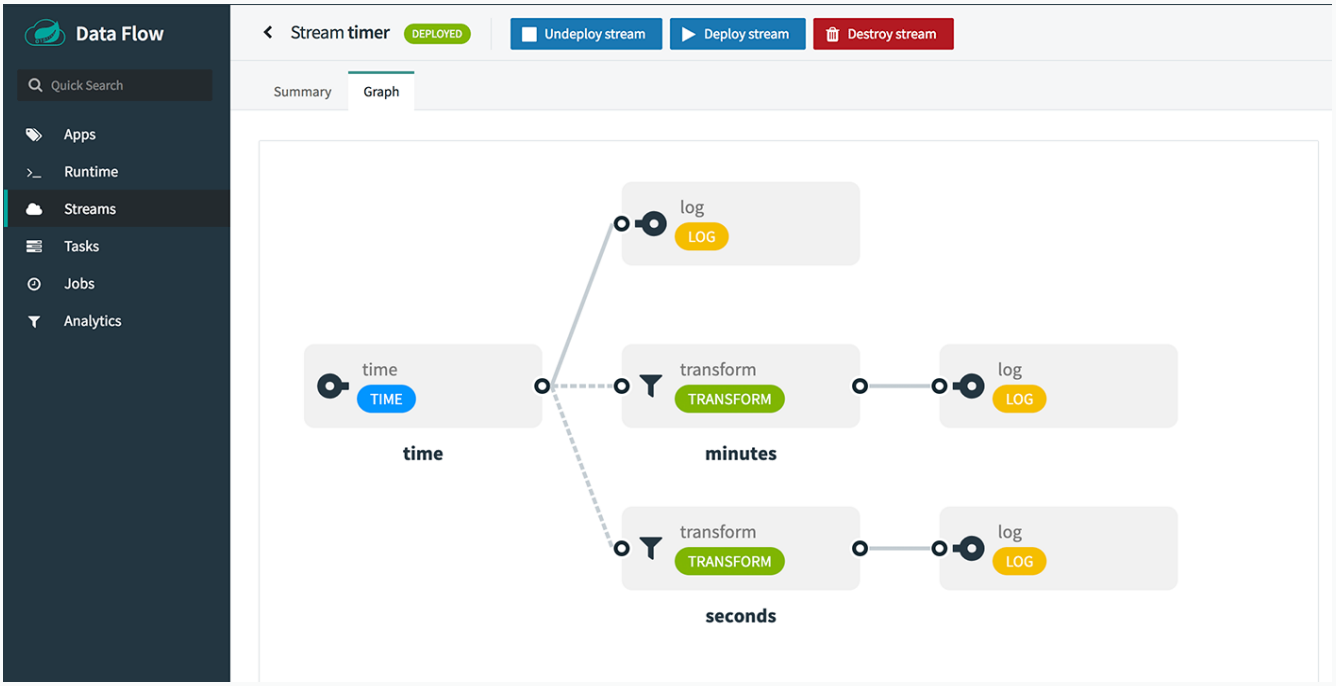
Spring Cloud Stream, Apache Kafka, RabbitMQ, Azure Event Hub ve diğer mesajlaşma sistemleriyle çalışırken üretkenliği arttırmak için üç temel soyutlama sağlar. Bu soyutlamalar bağlayıcılar, bağlamalar ve mesajlar olarak adlandırılır ve kaynak sağlama, içerik dönüştürme, hata işleme, yapılandırma yönetimi, tüketici grupları, bölümleme, izleme ve sağlık kontrolleri gibi özellikleri de destekler.

Spring Cloud Function, kodunuzu AWS, Azure gibi yerlerde çalıştırmanıza olanak tanır ve işlevleri zincirleyerek yeni yetenekler oluşturmanıza olanak sağlar. Bu araç, kapsamlı Spring API'lerini kullanarak üretkenliği artırır ve çoklu giriş ve çıkış desteği gibi özellikleri de destekler.

```
@SpringBootApplication
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }

    @Bean
    public  Function<String, String> uppercase() {
        return value → value.toUpperCase();
    }
}
```

Spring Cloud Data Flow, geliştiricilerin her türlü veri kaynağı ve hedefle çalışmak için araçlar ve otomasyonlar sunar. Yüksek verimli veri akışı dizileri oluşturmak, dağıtmak, yönetmek ve ölçeklendirmek için birden çok bulutta yerel platformda kullanılabilir. Bunun yanı sıra, uygulamaları kolayca oluşturmak ve izlemek için zengin bir kullanıcı arayüzüne sahiptir (Şekil 9).



Şekil 9. Spring Cloud Data Flow

Spring Cloud Stream, yalnızca Kafka Streams ile çalışmak için ikinci daha özel bir bağlayıcı sağlar. Bu bağlayıcı, Kafka'ya özgü özellikleri desteklemekle birlikte geliştirici üretkenliğine de odaklanır. Örneğin, KStream, KTable ve GlobalKTable gibi özellikleri destekler. Ayrıca, normal Spring Cloud Stream'deki gibi, bağlayıcı Kafka'ya bağlanmanın yanı sıra akışları ve konuları oluşturma, yapılandırma ve sürdürme ile de ilgilenir.

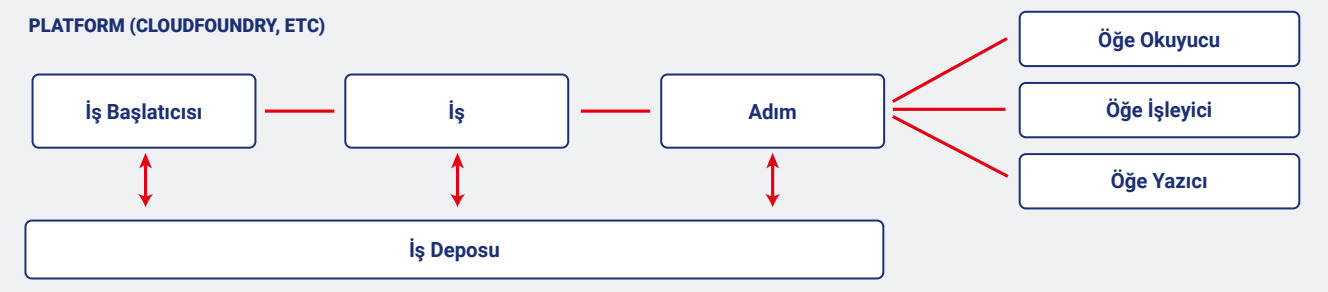
Spring AMQP ve Spring for Apache Kafka projeleri, Kafka veya RabbitMQ tabanlı mesajlaşma çözümleri için temel Spring kavramlarının uygulanmasını sağlar. Her ikisi de "şablon" (template) adı verilen üst düzey bir mesaj işleme soyutlaması ile bir "dinleyici konteyneri" (listener container) kullanarak mesaj odaklı POJO'ları destekler.

Uygulama entegrasyonu, her kuruluş için zor olabilir. Ancak, Spring Entegrasyonu popüler Spring programlama modelini en yaygın entegrasyon modellerini içerecek şekilde genişleterek bu yükü hafifletir. Mesajlaşma platformları, iletişim protokolleri, dosya sistemleri ve hizmet sağlayıcılar için hazır bağlayıcılar ve mesaj yönlendirme, veri dönüştürme ve filtreleme gibi yaygın kalıpların uygulamaları da vardır.

7. Toplu İş Uygulamaları (Batch)

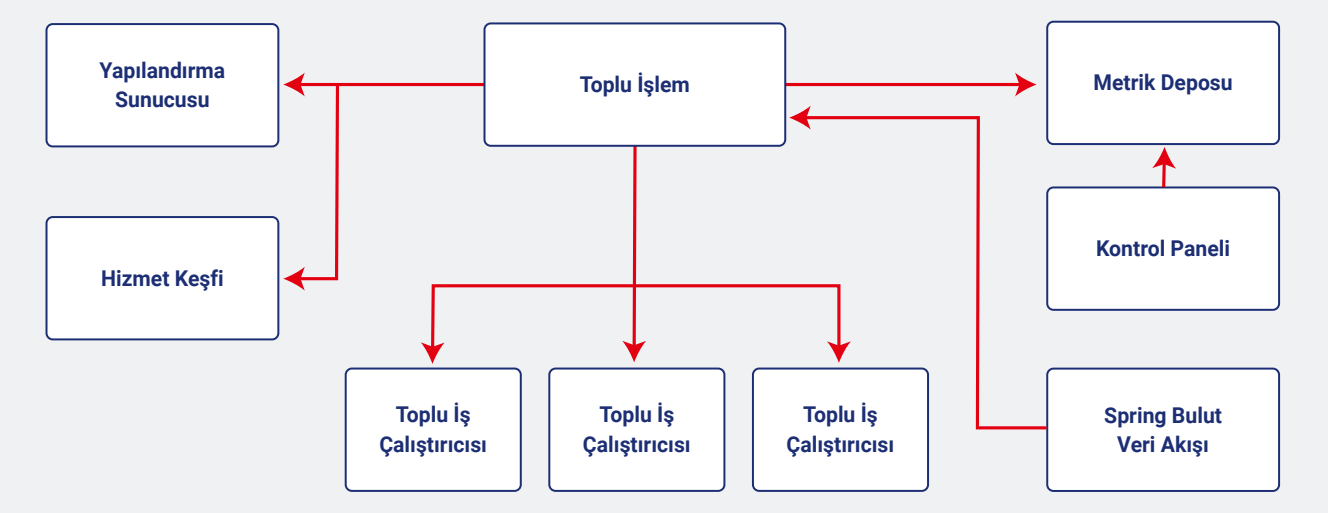
Birçok kullanım durumu için ideal hale getiren toplu işleme yeteneği, büyük miktarda veriyi verimli bir şekilde işlemek için önemlidir. Spring Batch, JVM'de toplu işler oluşturmayı mümkün kılarak endüstri standardı işleme kalıplarını uygular. Toplu işleme, harici etkileşim veya kesinti gerektirmeyen sonlu miktarda verinin işlenmesidir ve büyük miktarda veriyi işlemenin son derece verimli bir yoludur. SLA'lara dayalı olarak çalışmayı zamanlama ve önceliklendirme yeteneği, kaynakları en iyi şekilde kullanmaya olanak tanır.

Yaygın toplu iş modellerinin uygulanması, yüksek performanslı ve ölçeklenebilir toplu uygulamalar oluşturmanıza olanak tanır ve en kritik görev süreçleriniz için yeterince esnektir. Bölümleme ve yığın tabanlı işleme gibi toplu iş modelleri, Spring Boot'un üretim düzeyinde ek özellikleri ile toplu süreçlerin gelişimini hızlandırmaya olanak tanır (Şekil 10).



Şekil 10. Batch İşleme

Toplu işleme, özellikle bulut bilişim ve Hizmet Olarak Altyapı (IaaS) ile uyumlu bir teknolojidir. Bu teknoloji, uygulamaların isteğe bağlı, esnek ve hataya dayanıklı bir şekilde çalıştırılabilmesini sağlar. Spring Batch, bulut özellikleri kullanarak bu avantajlardan yararlanabilir. Ayrıca Spring Batch, diğer Spring API'leriyle kolay bir şekilde entegre olabilir ve veri okuma/yazma işlemleri için ItemReader ve ItemWriter, ilişkisel veritabanları ve NoSQL mağazaları için Spring Data desteği ve mesajlaşma için Apache Kafka ve RabbitMQ desteği gibi çeşitli özellikler sunar. Bu nedenle, Spring Batch birçok kullanım senaryosuna uygun bir toplu işleme çözümüdür (Şekil 11).



Şekil 11. Toplu İşlem Uygulaması

Sonuç ve Öneriler

Genel olarak, Spring Framework hala aktif olarak kullanılan bir teknolojidir ve özellikle büyük ölçekli projelerde tercih edilir. Ayrıca, diğer alternatif teknolojilerle karşılaştırıldığında, Spring Framework'ün daha uygun ve çok sayıda çözüm sunması ile avantaj sağladığı bilinmektedir.

Spring Framework; web uygulamaları, e-ticaret platformları, veritabanı işlemleri, güvenlik gibi farklı alanlarda kullanılabilmektedir. Bu çeşitlilik, Spring Framework'ün geniş bir kullanıcı kitlesine sahip olmasına ve birçok farklı sektörde tercih edilmesine olanak tanımaktadır.

Tüm bu özellikleri ve avantajları nedeniyle, Spring Framework dünya çapında birçok büyük şirket tarafından kullanılmaktadır. Bunlardan bazıları; Netflix, LinkedIn, PayPal, Amazon, IBM ve daha birçok önemli şirkettir.

Sonuç olarak; Spring Framework, Java uygulama geliştirme sürecini kolaylaştıran, verimli ve güvenilir bir platformdur. Ayrıca, açık kaynak olması ve büyük bir topluluğa sahip olması, kullanıcıların ihtiyaçlarını karşılamaları için yeterli kaynaklara sahip olmalarını sağlar. Tüm bunlar, Spring Framework'ün popülerliğini korumasına ve tercih edilen bir framework olmasına yardımcı olmaktadır.

Kaynakça

1. https://spring.io/
2. http://en.wikipedia.org/wiki/Spring_Framework
3. https://kerteriz.net/spring-framework-temelleri/
4. https://12factor.net/
5. https://snyk.io/blog/jvm-ecosystem-report-2018-platform-application/
6. https://www.jrebel.com/blog/2020-java-technology-report

TÜBİTAK

BİLGEM

YTE | YAZILIM TEKNOLOJİLERİ ARAŞTIRMA ENSTİTÜSÜ

İşçi Blokları Mahallesi Muhsin Yazıcıoğlu Caddesi No:51/C 06530 Çankaya/ANKARA

+90 (312) 289 92 22 - yte.bilgem@tubitak.gov.tr

TÜBİTAK - BİLGEM Yazılım Teknolojileri Araştırma Enstitüsü (YTE)

yte.bilgem.tubitak.gov.tr