**SPRİNG-CORE**

**BİR SINIF GELİŞİME AÇIK, DEĞİŞİME KAPALI OLMALIDIR.**

**Spring nedir :** Spring java tabanlı web uygulamaları geliştirirken , daha hızlı daha kolay, ve daha profesyonel uygulamalar ortaya çıkarmamızı sağlar.

Mvc,orm , veritabanı,güvenlik,test cache gibi her alana elini atmıştır.

**->**Java ile geliştirilmiş yüzlerce framework vardır bunlardan en meşhuru 2 tanesi:

1-Spring

2-Hibernate

**->**Bir şey apache lisansına sahipse ücretsizdir.(Spring ücretsizdir).

Genel kültür : Sun firması 1998-2000 yılları arasında kurumsal uygulamaları daha hızlı geliştirmek için çok büyük yatırım yapmış fakat istediğini sonucu alamayarak kendi sonunu hazırlamış.

2008 yılı gibi ise Javanın sahibi olan Sun firması , Javayı Oracle firmasına satmıştır.

**->**Spring’in bir çok modülü vardır ve proje geliştirirken bütün modülleri kullanmak zorunda değiliz . İhtiyacımıza göre modüllerimizi ekleyip kullanabiliriz.

**->**Spring birbirinden bağımsız olarak kullanabileceğimiz modüllerden(parçalardan) oluşur.Sadece SPRİNG CORE denilen çekirdek modülü kullanmak zorundayız diğerlerini kullanmak bizim tercihimize göre.

**->**Spring’I diğer frameworklerle beraber kullanabiliriz bu da çok önemli.(spring frameworku masa gibi düşün üzerine her şey koyabilirsin.)

EJB(Enterprise JavaBeans) : Sun firması tarafından kurumsal projelerin ortak sorunlarına çözüm olmak için ortaya çıkmıştır.

Birçok alanda kullanımı çok etkiliymiş fakat kullanımı zahmetli ve öğrenilmesi zor olduğu için sektörde cazibesini yitirdi . O koltuğu da SPRİNG aldı , Spring dedi ben daha kolay bir şekilde istediklerinizi sunuyorum gelin beni kullanın.

Daha sonra EJB kendisini geliştirdiği şuanda 2 side(EJB-Spring) aynı kefede ikisini de kullanabiliriz. Fakat iş ilanlarını ve sektörde ki proje ve kaynak sayısı düşünüldüğü zaman Spring Framework öğrenmek en mantıklısı…

Spring tarafından yönetilen her bir nesneye bean denir.Bean aslında sıradan java classlarıdır.

Spring Container(kap) : Aslında şöyle düşünürsek dünya bir containerdır(kaptır.)

Çükü içinde milyarlarca insan, canlı, bitki nesne tutabilir.

Spring’in de bi container yani kabı var ve bu kap içerisinde spring benim java beanlarımı nesnelerimi tutuyor ,aralarındaki ilişkileri sağlıyor.

Kısacası benim java beanlarımı depoladığım mekan diyebilirim.

Context Nedir ? : Context , bizim spring container içerisinde tanımlanmış olan beanlarımız arasındaki bağı, ilişkiyi sağlayan , bilgileri tanımlayan , classlardan beanları oluşturan,aralarında ki ilişkileri kuran yapıya context deniyor kısacası…

İnversion Of Control(IoC) : Controlün springe cevrilmesi , bütün yükümlülüğün ve sorumluluğun ona verilmesi demektir. Artık beanlarım spring tarafından oluşturulur,yönetilir bütün top springtedir.

Dependecy Injection(bağımlılıkların enjeksiyonu) : Spring container içerisinde tanımlanmış olan beanlarımı projemde istediğim class veya metot içerisine enjekte etmeme ,set etmeme yarar.

Composition(komposizyon) : Bir sınıf diğer sınıfı miras almadan metotlarını ,propertylerini kullanabiliyorsa buna composition denir.

Bu nesneler birbirine göbekten bağlı olmuş. Miras almadan kullandığımız sınıfın metotunu değiştirirsek onu kullanan bütün sınıflar patlar buda kötü bir şeydir.

Dependency Enjection yapmanın 3 yolu vardır;

**Aslında muhabbet şu , biz spring context’teki yani xml dosyamızda spring tarafından oluşturulan spring beanlardan constructor veya setter metotları kullanarak atanan değerleri java tarafından yakalıyoruz . Binevi enjekte etmiş oluyoruz aslında.**

1-setter method

2-constructor

3-sıradan bir method

**@Autowire**(otomatik bağlama) kullanırken 3 tipte bağlama yapılabilir;

1-ByName

2-ByType

3-constructor

**ByName**: eğer ki byname **yaparsak bean’ın id değeri ile class içinde ki referans ismi aynı** olursa bağlar. Olmazsa bağlama yapılmaz.

Bağlanamktan kasıt bean’I alıp referansa set etmek , atamak demektir.

**ByType**: **Class içindeki referansın tipi ile ,tanımlanan bean’ın tipi aynı ise birbirlerine** otomatik olarak bağlar.

**Constructor:** bean ile class içindeki referans’I constructor aracılığıyla otomatik olarak bağlar yani set eder , atar.

**@Qualifier** : applicationContext.xml içerisinde aynı tipte birden fazla bean var ise,

Biz onu @Autowire ile bağlarken hangisini kullanıcaksak,seçiceksek @Qualifier anatasyonu ile bunu belirtiyoruz.

**@PostConstruct :** Bean oluşturulmadan önce tetiklenmesini istediğimiz metotu çalıştırmak için kullanırız.

(xml ile yaparken init-method attribute’sini kullanıyorduk)

**@PreDestroy :** Bean oluşturulup öldükten sonra ,bellekten düştükten sonra tetiklenmesini istediğimiz metot’u belirtmek için kullanıırız.

(xml ile yaparken destroy-method attirubute a denk geliyor.)

**@Component :** Bu anatasyon ile tanımlanan sınıflar xml veya appconfig içersiinde beanları oluşturuyor otomatik olarak. Ben görmessemde varlığından haberdarım.