Title

JDBC vs. JDBC Template vs. Hibernate

. . .

Merhaba, bu yazımda JDBC, JDBCTemplate ve Hibernate'in ne olduğunu ve aralarındaki farkları açıklamaya çalışacağım. Hadi başlayalım.

. . .

JDBC Nedir?

Java JDBC (Java DataBase Connectivity) MySQL, Oracle, MSSQL vb. veritabanlarına bağlanmak, veri çekmek, listelemek , eklemek, silmek, güncellemek gibi işlemleri yapmak için kullandığımız pakettir.

JDBC yapısı veritabanından bağımsız olduğundan dolayı SQL destekleyen tüm ilişkisel veritabanları ile birlikte çalışır.

JDBC kullanmak için gerekli adımlar;

- Bağlantı yapmak istediğimiz veritabanına ait driver'ın eklenmesi
- Veritabanı bağlantısı
- SQL sorgu cümlesinin yazılması
- Sonuçların alınması

. . .

JDBC Template Nedir?

Veriler ile iletişime geçmek için işlemlerimizi normal şartlar altında JDBC kullanarak hallederiz. Fakat, JDBC ile işlemlerimizi gerçekleştirirken bağlantı bilgileri tanımlayarak her veritabanı işlemlerinde bağlantı açmak, bittiğinde bağlantıyı kapatmak gibi zorunda olduğumuz işlemleri defalarca tekrar etmek bizim için bir süre sonra zahmetli bir hal alabiliyor.

Spring JDBC Template tam bu noktada devreye giriyor. Bizi asıl yapmamız gereken işe yoğunlaştırırken, veritabanı işlemleri için önceden yaptığımız tekrarlı işlemleri en aza indirip "otomatik" olarak yapılmasını sağlayabiliyor.
Yani bizleri JDBC'e göre daha pratik bir yola sokuyor diyebiliriz.

. . .

HIBERNATE Nedir?

Hibernate bir ORM Framework'üdür. 2001 yılında Gavin King tarafından open-source olarak yayınlandı. Hem Java hem de C# da kullanılabilen bir framework'dür.

Hibernate ile yapılan tüm işlemleri JDBC ile yapabilmekteyiz, ancak kod karışıklığı yaratmaması ve OOP mantığına uygun bir yazılım standartına ulaşmak için programlarda veritabanlarına ulaşmak için Hibernate tercih ederiz. Bizi yazılım karmaşıklığından kurtarır.

Hibernate kullanırken Plain Old Java Object (POJO) adı ile bilinen Java nesnelerine ihtiyacımız vardır. POJO bir bakıma Bean'dir. Hibernate'in hangi veritabanına nasıl işleneceği XML dosyasında belirtilir.

. . .

Aralarındaki Farklar Nelerdir?

Bazı durumlarda doğrudan schema ile çalışmak gerekli olacaktır.
Örneğin, yapısı bilinmeyen bir tablodan veri okumamız gerekir. JDBC bu noktada biraz acı verici olabilir ve ayrıca Hibernate, veritabanı kavramlarını ve öğelerini gizlemeye çalışır. Bu noktada Spring bizlere, JDBC'yi dahili olarak kullanan ve kolaylık sağlayan JDBC Template'i sunar. JDBC Template ile JDBC'nin gücünü elde ediyoruz ancak boiler-plate kod yazmaya gerek kalmıyor.

Spring JDBC Template, standart JDBC'ye göre aşağıdaki avantajlara sahiptir :

- Veritabanı bağlantılarını otomatik olarak açıp kapatma mekanizmasına sahiptir.
- JDBC SQLException'larını RuntimeExceptions'a dönüştürerek programcının hatalara daha esnek tepki vermesini sağlar.
- Hata mesajlarını daha anlaşılır hale dönüştürür.
- SQL sorgularını doğrudan yazmak için methodlar sağlar.

Hibernate ise bizlere şu avantajları sağlar:

- LGPL lisansı altında açık kaynak kodlu ve hafiftir.
- Performansı hızlıdır çünkü önbellek Hibernate'de dahili olarak kullanılır.
- HQL (Hibernate Query Language), SQL'in nesne yönelimli sürümüdür. Veritabanından bağımsız sorgular üretir. Böylece veritabanına özel sorgular yazmanıza gerek kalmaz. Hibernate öncesinde, proje için veritabanı değiştirilirse, bakım sorununa yol açan SQL sorgusunu da değiştirmemiz gerekir.
- Veritabanı tablolarını otomatik olarak oluşturma olanağı sağlar. Bu nedenle veritabanında manuel olarak tablo oluşturmaya gerek yoktur.
- Birden çok tablodan veri almak kolaydır.
- Sorgu önbelleğini destekler, sorgu ve veritabanı durumu hakkında istatistikler sağlar.

. . .

Umarım bu makale Java geliştiricilerine bir nebze de olsa katkı sağlamıştır.

Başka bir yazıda görüşmek üzere.

İstek ve önerileriniz için bana batuhankiltac@gmail.com adresinden ulaşabilirsiniz.