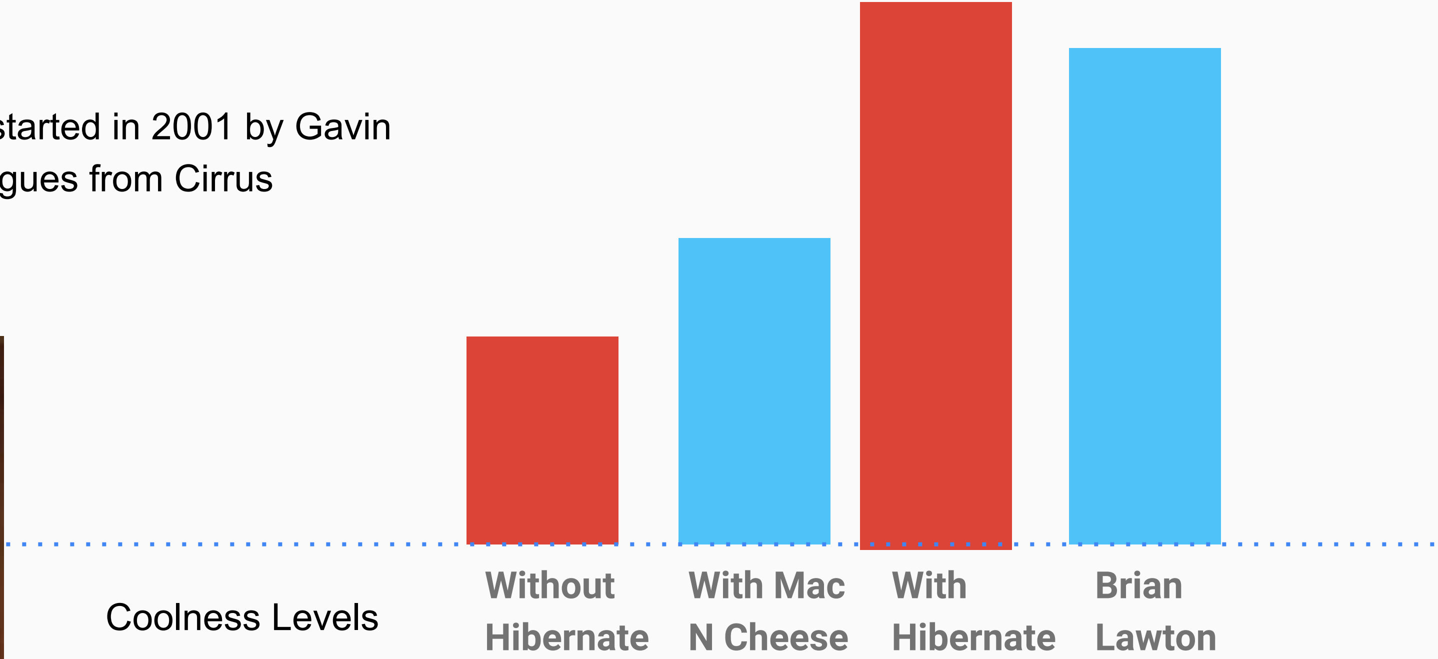
HIBERNATE 4.3.6

By:
Omar Salih, Michael Lacy, Sam Luebbers
Kyle Hopfer, Chau Tran

Who made Hibernate

Who made it?

Hibernate was started in 2001 by Gavin King with colleagues from Cirrus Technologies



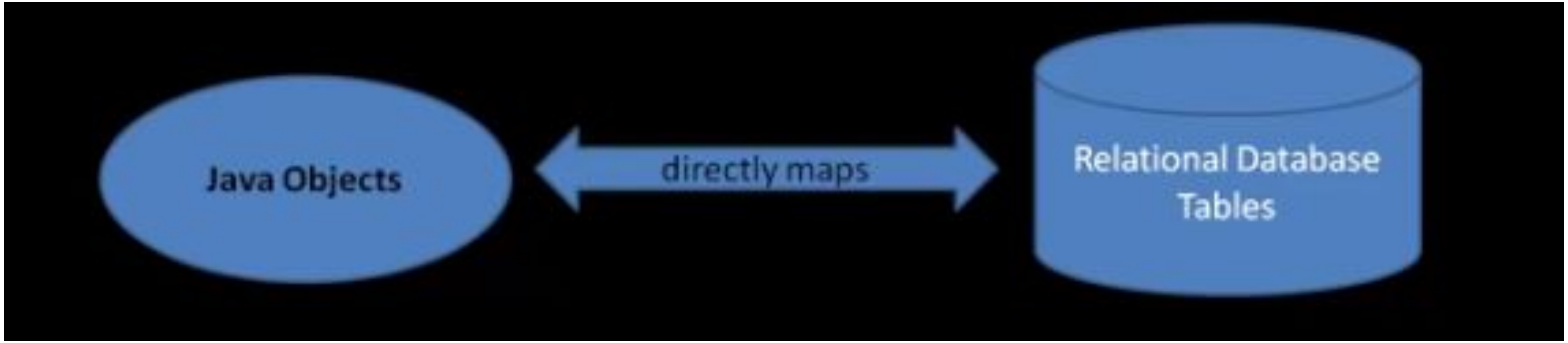
What is Hibernate???

It’s a free ORM that is an object-relational mapping framework for Java language. Hibernate is an object-oriented framework that allows you to map the object oriented model of a POJO (Plain Old Java Object) class onto a relational database.

ADVANTAGES:

- Hibernate takes care of mapping Java classes to database tables using XML files and without writing any lines of code.
- Provides simple APIs for storing and retrieving Java objects directly to and from the database.
- If there is change in Database or in any table then the only need to change XML file properties.
- Abstract away the unfamiliar SQL types and allows us to work around familiar Java Objects.
- Hibernate does not require an application server to operate.
- Manipulates complex associations of objects of your database.
- Minimizes database access with smart fetching strategies.
- Provides simple querying of data.

Usefulness of Hibernate



Hibernate maps Java classes to database tables and from Java data types to SQL data types

Supported Databases

Following is list of few of the database engines supported by Hibernate. Hibernate supports almost all the major RDBMS.

- HSQL Database Engine
- DB2/NT
- MySQL
- PostgreSQL
- FrontBase
- Oracle
- Microsoft SQL Server Database
- Sybase SQL Server
- Informix Dynamic Server

Supported Technologies

Hibernate supports a variety of other technologies, including the following:

- XDoclet Spring
- J2EE
- Eclipse plug-ins
- Maven

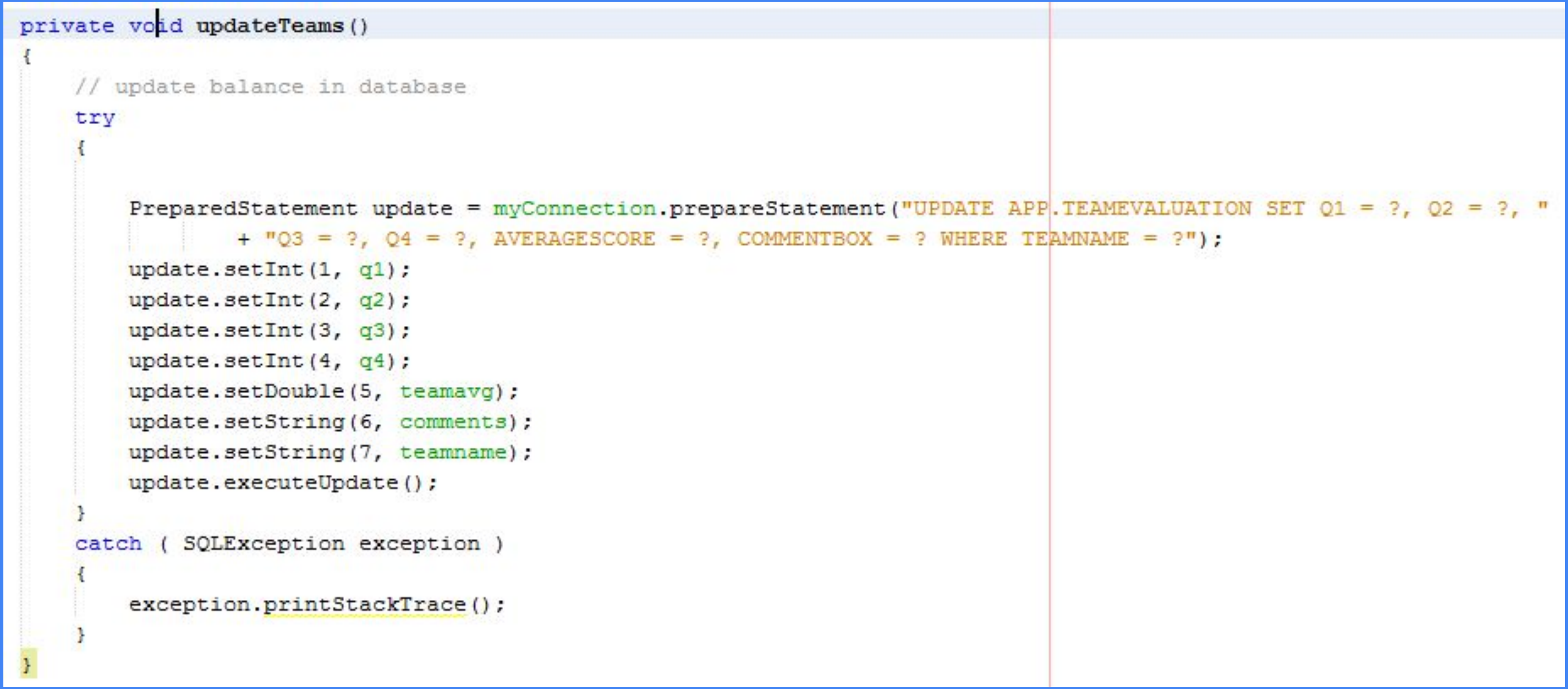
What does Hibernate do???

It allows you to grow without having to write much more code. Big projects and big databases, doesn't need sql.

Required Files for Hibernate

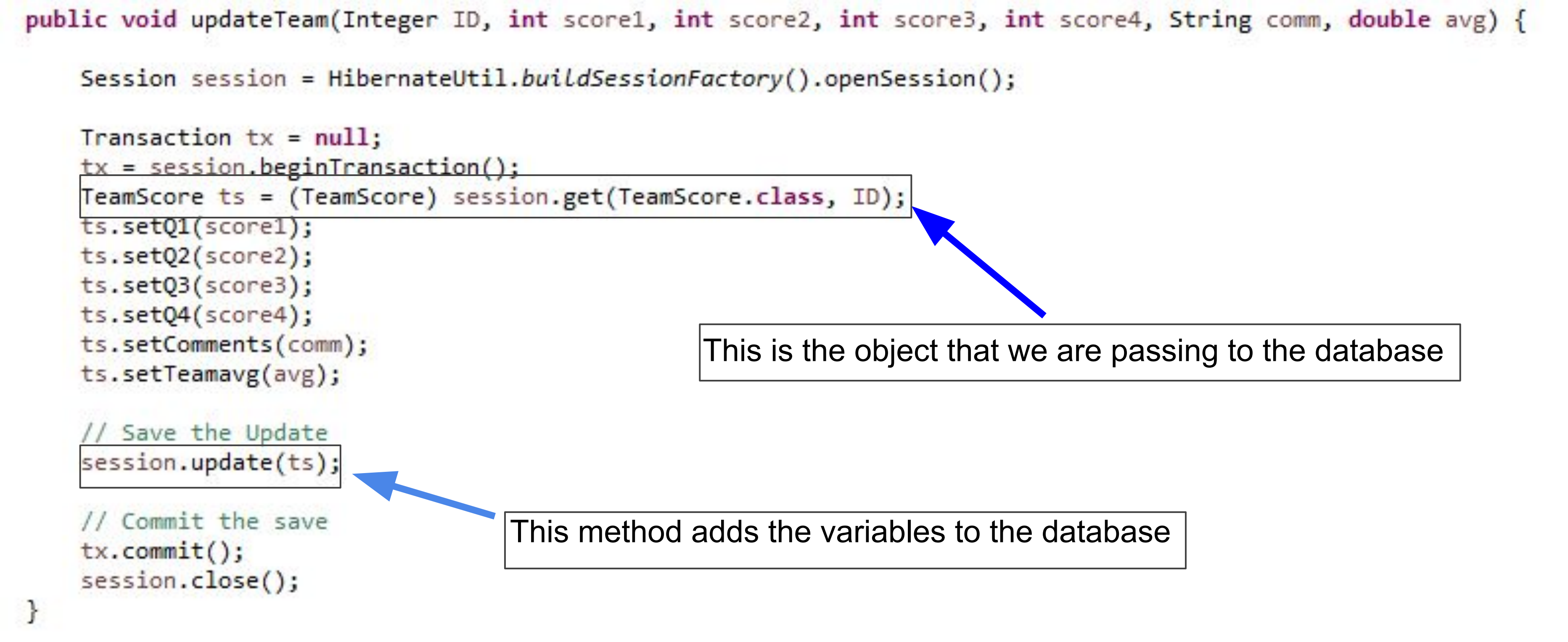
- Main Class
- POJO
- Configuration File, commonly named hibernate.cfg.xml

UpdateTeams() Method Without Hibernate



```
private void updateTeams()
{
    // update balance in database
    try
    {

        PreparedStatement update = myConnection.prepareStatement("UPDATE APP.TEAMEVALUATION SET Q1 = ?, Q2 = ?, "
                + "Q3 = ?, Q4 = ?, AVERAGESCORE = ?, COMMENTBOX = ? WHERE TEAMNAME = ?");
        update.setInt(1, q1);
        update.setInt(2, q2);
        update.setInt(3, q3);
        update.setInt(4, q4);
        update.setDouble(5, teamavg);
        update.setString(6, comments);
        update.setString(7, teamname);
        update.executeUpdate();
    }
    catch ( SQLException exception )
    {
        exception.printStackTrace();
    }
}
```

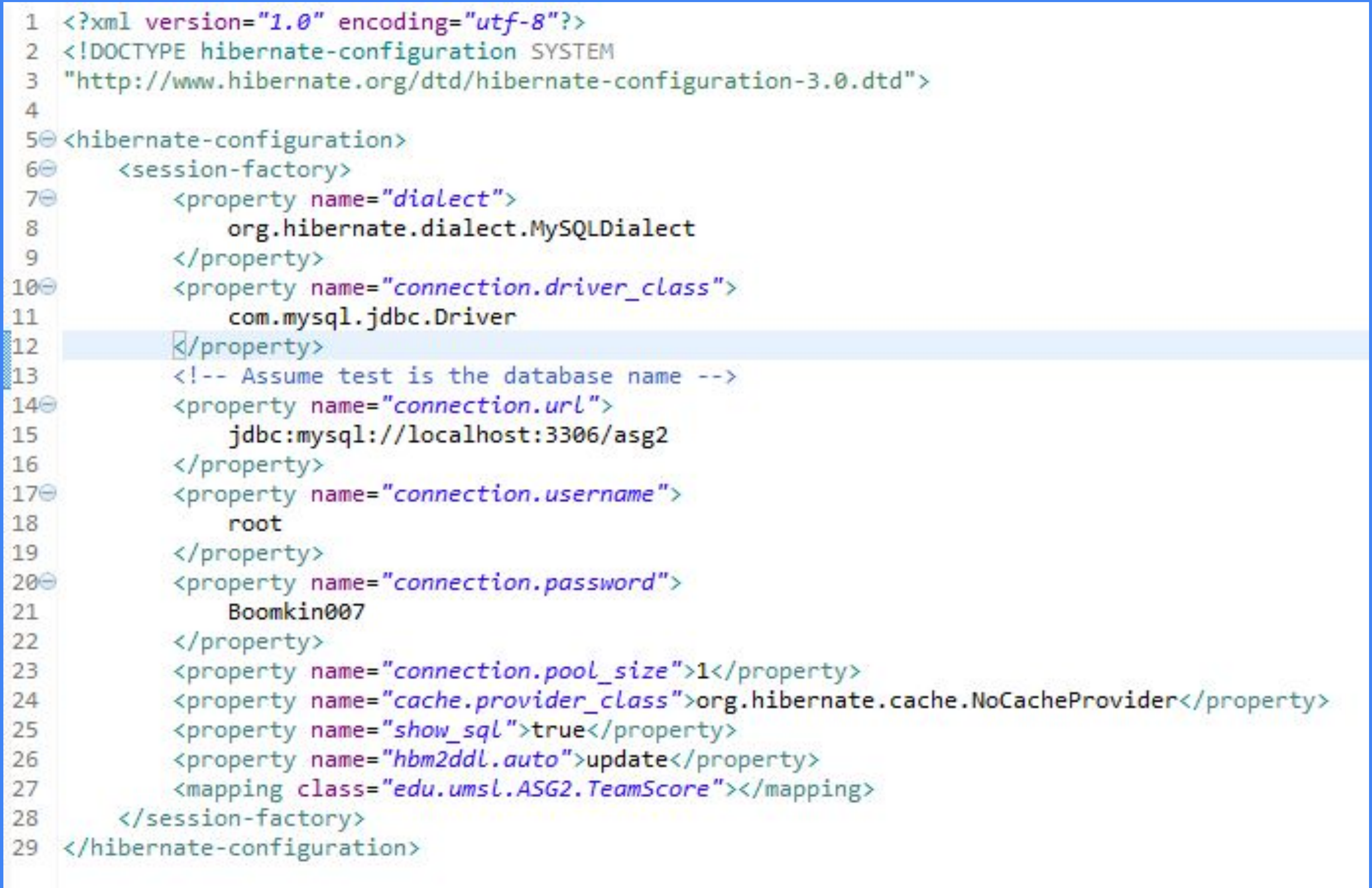
UpdateTeams() Method With Hibernate



```java
public void updateTeam(Integer ID, int score1, int score2, int score3, int score4, String comm, double avg) {

    Session session = HibernateUtil.buildSessionFactory().openSession();

    Transaction tx = null;
    tx = session.beginTransaction();
    TeamScore ts = (TeamScore) session.get(TeamScore.class, ID);
    ts.setQ1(score1);
    ts.setQ2(score2);
    ts.setQ3(score3);
    ts.setQ4(score4);
    ts.setComments(comm);
    ts.setTeamavg(avg);

    // Save the Update
    session.update(ts);

    // Commit the save
    tx.commit();
    session.close();
}
```

This is the object that we are passing to the database

This method adds the variables to the database

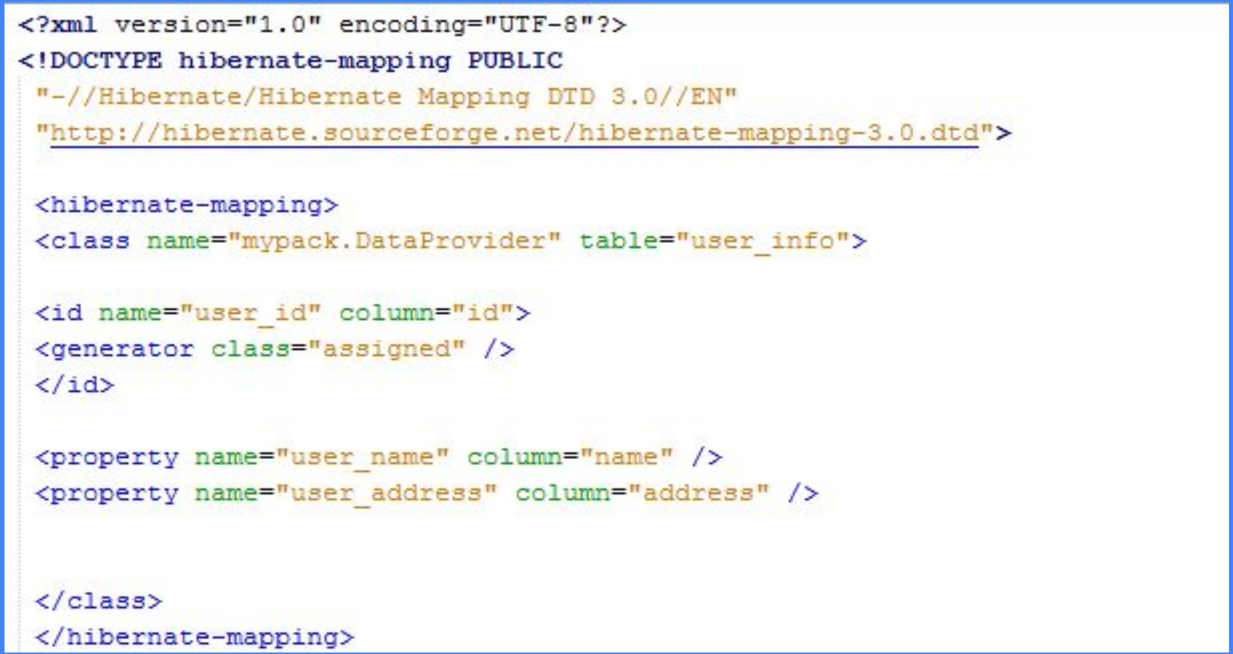


```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration SYSTEM
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="dialect">
            org.hibernate.dialect.MySQLDialect
        </property>
        <property name="connection.driver_class">
            com.mysql.jdbc.Driver
        </property>
        <!-- Assume test is the database name -->
        <property name="connection.url">
            jdbc:mysql://localhost:3306/asg2
        </property>
        <property name="connection.username">
            root
        </property>
        <property name="connection.password">
            Boomkin007
        </property>
        <property name="connection.pool_size">1</property>
        <property name="cache.provider_class">org.hibernate.cache.NoCacheProvider</property>
        <property name="show_sql">true</property>
        <property name="hbm2ddl.auto">update</property>
        <mapping class="edu.umsl.ASG2.TeamScore"></mapping>
    </session-factory>
</hibernate-configuration>
```

Mapping File vs. Annotations

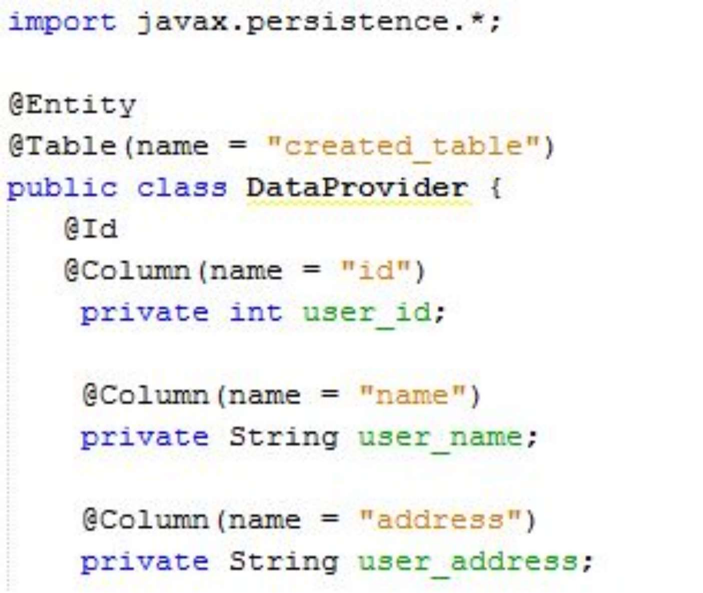
Making a mapping file forces you to write more XML, annotations don’t

Mapping File Example



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
 "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
 "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

 <hibernate-mapping>
 <class name="mypack.DataProvider" table="user_info">

 <id name="user_id" column="id">
 <generator class="assigned" />
 </id>

 <property name="user_name" column="name" />
 <property name="user_address" column="address" />


 </class>
 </hibernate-mapping>
```

Annotations Example



```
import javax.persistence.*;

@Entity
@Table(name = "created_table")
public class DataProvider {
   @Id
   @Column(name = "id")
    private int user_id;

    @Column(name = "name")
    private String user_name;

    @Column(name = "address")
    private String user_address;
```

Let’s run this bad boy!

WRAP
IT
UP

QUESTIONS?

References:

- http://www.gontu.org/hibernate-framework-tutorials/
- http://www.tutorialspoint.com/hibernate/index.htm
- https://en.wikipedia.org/wiki/Hibernate_%28framework%29
- http://hibernate.org/