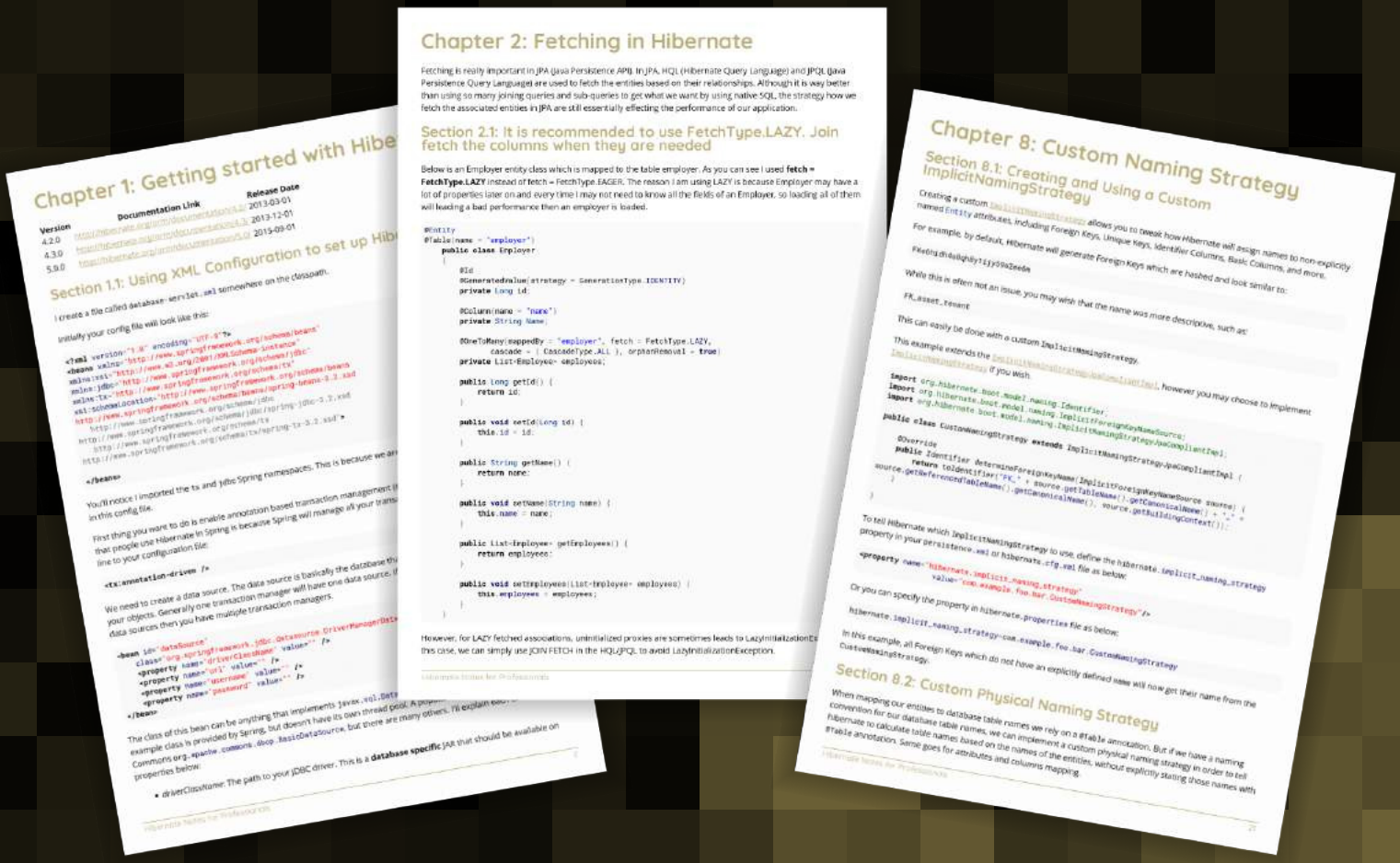
Hibernate

Notes for Professionals



30+ pages

of professional hints and tricks

Contents

About

Please feel free to share this PDF with anyone for free,
latest version of this book can be downloaded from:
https://goalkicker.com/HibernateBook

Please send feedback and corrections to web@petercv.com

Chapter 1: Getting started with Hibernate

Version	Documentation Link	Release Date
4.2.0	http://hibernate.org/orm/documentation/4.2/	2013-03-01
4.3.0	http://hibernate.org/orm/documentation/4.3/	2013-12-01
5.0.0	http://hibernate.org/orm/documentation/5.0/	2015-09-01

Section 1.1: Using XML Configuration to set up Hibernate

I create a file called `database-servlet.xml` somewhere on the classpath.

Initially your config file will look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:jdbc="http://www.springframework.org/schema/jdbc"
xmlns:tx="http://www.springframework.org/schema/tx"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
   http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-3.2.xsd
   http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-3.2.xsd">

</beans>
```

You'll notice I imported the `tx` and `jdbc` Spring namespaces. This is because we are going to use them quite heavily in this config file.

First thing you want to do is enable annotation based transaction management (`@Transactional`). The main reason that people use Hibernate in Spring is because Spring will manage all your transactions for you. Add the following line to your configuration file:

```
<tx:annotation-driven />
```

We need to create a data source. The data source is basically the database that Hibernate is going to use to persist your objects. Generally one transaction manager will have one data source. If you want Hibernate to talk to multiple data sources then you have multiple transaction managers.

```
<bean id="dataSource"
    class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="" />
    <property name="url" value="" />
    <property name="username" value="" />
    <property name="password" value="" />
</bean>
```

The class of this bean can be anything that implements `javax.sql.DataSource` so you could write your own. This example class is provided by Spring, but doesn't have its own thread pool. A popular alternative is the Apache Commons `org.apache.commons.dbcp.BasicDataSource`, but there are many others. I'll explain each of the properties below:

- *driverClassName*: The path to your JDBC driver. This is a **database specific** JAR that should be available on