Hibernate Tip: The best way to persist a ZonedDateTime

Question:

During last week's Hibernate workshop, I got a few questions about Hibernate's handling of *ZonedDateTime* objects. To sum them up, all attendees wanted to know what's the best way to persist an attribute of type *ZonedDateTime* with Hibernate.

Solution:

Since version 5, Hibernate supports some classes of the Date and Time API as basic types. The *ZonedDateTime* class is one of them.

```
@Entity
public class Review {

    private ZonedDateTime postedAt;

    ...
}
```

But it's not supported in the way you might expect. Hibernate doesn't persist any timezone information for it. It converts the value to the local timezone and stores it as a timestamp. My current timezone is UTC+2. So, if I persist a *ZonedDateTime* in EST, which is UTC-4, Hibernate converts it to UTC+2 before it writes it to the database.

As I explained previously, this causes problems when:

- You use a time zone with daylight saving time
- Your systems use a different time zones
- You change the timezone of your application for some reason

That's why you should use the configuration parameter *hibernate.jdbc.time_zone*, which Hibernate introduced in version 5.2. It enables you to specify the timezone that Hibernate shall use for its conversions. In this example, I set the timezone to UTC.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
version="2.1"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistenc
e
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd"
>
    <persistence-unit name="my-persistence-unit">
        <description>Hibernate Tips</description>
        <provider>org.hibernate.jpa.HibernatePersistenceProvid
er</provider>
        <exclude-unlisted-classes>false</exclude-unlisted-
classes>


        <properties>
            <property name="hibernate.dialect"
value="org.hibernate.dialect.PostgreSQLDialect" />
            <property name="hibernate.jdbc.time_zone"
value="UTC"/>


            <property name="javax.persistence.jdbc.driver"
value="org.postgresql.Driver" />
```

Hibernate Tip: The best way to persist a ZonedDateTime

```
<property name="javax.persistence.jdbc.url"
value="jdbc:postgresql://localhost:5432/recipes" />
        <property name="javax.persistence.jdbc.user"
value="postgres" />
        <property name="javax.persistence.jdbc.password"
value="postgres" />
      </properties>
   </persistence-unit>
</persistence>
```

If you now persist a new entity, Hibernate tells you in the log messages that it persists the timestamp with the timezone specified by the *LocalDate*. In this example, that's UTC-4.

```
EntityManager em = emf.createEntityManager();
em.getTransaction().begin();

// Persist a new Review using a timezone UTC-4
Review r = new Review();
r.setComment("Amazing Book!");
r.setPostedAt(ZonedDateTime.now(ZoneId.of("UTC-4")));
em.persist(r);

em.getTransaction().commit();
em.close();
```

```
06:30:59,636 DEBUG [org.hibernate.SQL] - select nextval
('hibernate_sequence')
06:30:59,638 DEBUG [org.hibernate.SQL] - insert into
Review (fk_book, comment, postedAt, id) values (?, ?, ?, ?)
06:30:59,638 TRACE
[org.hibernate.type.descriptor.sql.BasicBinder] - binding
parameter [1] as [BIGINT] - [null]
06:30:59,638 TRACE
[org.hibernate.type.descriptor.sql.BasicBinder] - binding
parameter [2] as [VARCHAR] - [Amazing Book!]
06:30:59,639 TRACE
[org.hibernate.type.descriptor.sql.BasicBinder] - binding
parameter [3] as [TIMESTAMP] - [2018-05-01T00:30:59.634-
04:00[UTC-04:00]]
06:30:59,640 TRACE
[org.hibernate.type.descriptor.sql.BasicBinder] - binding
parameter [4] as [BIGINT] - [2]
```

But when you take a look at the database record, you can see that it converted it to UTC.

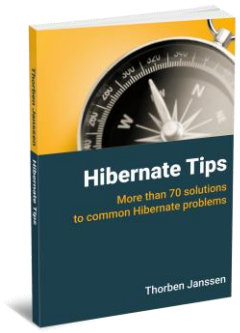
	id [PK] bigint	comment character varying (255)	postedat timestamp without time zone	book_id bigint
1	2	Amazing Book!	2018-05-01 04:35:52.645	2

Learn more

If you're using the Date and Time API, you might also be interested in these articles:

- Hibernate 5: How to persist LocalDateTime & Co with Hibernate
- How to persist LocalDate and LocalDateTime with JPA 2.1
- How To Map The Date And Time API with JPA 2.2
- Hibernate Tips: How to map a java.util.Date to a database column

Hibernate Tips Book



Get more recipes like this one in my book Hibernate Tips: More than 70 solutions to common Hibernate problems.

It gives you more than 70 ready-to-use recipes for topics like basic and advanced mappings, logging, Java 8 support, caching and statically and dynamically defined queries.